গ্নু-লিনাক্স ইশকুল

জিএলটি মধ্যমগ্রাম
glt-mad@ilug-cal.org

শূন্য নম্বর দিনে আমরা হাতের সামনে একটা আস্ত পিসি পেয়ে নাড়াচাড়া শুরু করেছিলাম, ভৌত উপাদান নিয়ে একদম প্রাথমিক দু-চারটে কথা, তারপর একটা কম্পিউটারের তথ্য নাড়াঘাঁটা ও জমানোর তরকিব নিয়ে কিছু কথা বলে শেষ করেছিলাম। এক নম্বর দিনে আমরা শুরু করেছিলাম একটা কূট এবং আপাতঅগ্রাহ্য সেকশন দিয়ে, যাতে সিপিইউ-র রেজিস্টার কাঠামো এবং মেমরি পড়ার সিপিইউ-প্রথা নিয়ে কথা বলেছিলাম, তারপর গেছিলাম একটা কম্পিউটারের সফটওয়ার অংশটার একটা কাঠামো খাড়া করার কাজে। প্রোগ্রাম, সফটওয়ার, অ্যাপ্লিকেশন — এবং এদের সামগ্রিক আধার — মানে অপারেটিং সিস্টেম। অপারেটিং সিস্টেমের ভিতর সফটওয়ারের বিভিন্ন ধারাগুলোর, কমান্ড ইন্টারপ্রিটার, এডিটর, কম্পাইলার, এবং অ্যাপ্লিকেশন — এদের ঠিক ভূমিকাটা কী সেই বিষয়ে একটা আন্দাজ দিয়েছিলাম। আজ, মানে জিএলটি-ম্যাড পাঠমালার দুই নম্বর দিনে অপারেটিং সিস্টেমের ভূমিকা নিয়ে একটু কথা বলে চলে যাব, সিপিইউর পরই কম্পিউটারের সবচেয়ে গুরুত্বপূর্ণ দিক — আইও মানে ইনপুট/আউটপুট নিয়ে প্রাথমিক কিছু কথায়। এই পরের অংশের অনেকটাই ঠিক এক নম্বর দিনের এক নম্বর সেকশনের মত কূট এবং আপাতঅগ্রাহ্য। পরে যখন শেল কারনেল নিয়ে দ-য়ে পড়বেন তখন বুঝবেন কত কাজের ওগুলো, ফিরে আসতে হবে। এই পাঠমালার মূল পরিকল্পনাটা প্রতি মুহূর্তে আমূল বদলে যাচ্ছে, খসড়াগুলো রয়েছে নিরন্তর প্রব্রজ্যায়, এই বদল ঘটানোর চক্করে ভাঙা ঠ্যাং-এর প্লাস্টারের নিচে হিলের উপর টাল খেতে খেতে আমায় বারকয়েক করে ঠাকুরঘরে যেতে হচ্ছে, এক্সট্রা, ঠাকুরঘরে ছাড়া আমি আবার খুব মন দিয়ে কিছু ভাবতে পারিনা। ঠাকুরঘর শব্দটা, প্লিজ, এখানে একটু ভিন্ন কনটেক্সটে পড়বেন — যে কলোনিয়াল সভ্যতার দায় নিয়ে রামরাম বসু মশাই তখন সেবারত, আমাদের আদি পরিত্রাতাতাদের প্রমুখ উইলিয়ম কেরিকে, তখন শীতের ভোরের কুয়াসা ফেটে আলো ক্রমে আসিতেছে, রক্তিমতা প্রভাব বিস্তার করিবে, আমরা কলোনির প্রাকৃতজনেরা পশ্চিমের সভ্যতার পুঁজির মাইক্রোসফটের দিকে ক্রমে অগ্রসরমান পুষ্পের উষ্ণতায় চিহ্নিত হইব, নদীবুকে ভাসমান বজরায়, নদীতটরেখা জুড়ে সারিসারি উপাসীন মানবমিছিল দেখে কেরি প্রশ্নশীল, প্রতিটি উপাসীন মানুষের সামনে উপাসনার উপচার, উজ্জ্বল ধাতুপাত্রে পূতবারি, রামো রামো, রামরাম বললেন, দে আর প্রেয়িং টু দেয়ার গড স্যার, আমিও তো আজো বহন করে চলেছি সেই কলোনিয়ালতার সভ্যতার আলোকপ্রাপ্তির উত্তরাধিকার।

।। দিন দুই।।

১।। দুটো কাজ : করছে এক জন

আগের দিন অপারেটিং সিস্টেম নিয়ে বেশ কিছু কথা বলার পরও, আজ যদি প্রশ্ন করা যায়, অপারেটিং সিস্টেম বস্তুটা ঠিক কী — খায় না মাথায় দেয় — আমাদের খুব স্পষ্ট একমুখী একটা দ্বর্থহীন উত্তর দিতে বেশ ব্যথা হবে। শুধু তাই নয়, অনেকদিন ধরে কম্পুটার ব্যবহার করার পর নিজের থেকেই বেশ অনুভব করা যায় কাজের ক্ষেত্রের গোটাটা জুড়ে অপারেটিং সিস্টেমের সর্বব্যাপী সর্বত্রগামী সর্বশক্তিমান উপস্থিতি, কেউ একটা অন্তরালে বসে সব কিছু গুছিয়ে যাচ্ছে। সুগৃহিণীপনার হদ্দমুদ্দ একেবারে। কে কী করতে পারে, করতে চাইতে পারে, সচরাচর করে থাকে, কে কে কখন কখন কোথায় কোথায় কী করছে, এই করার সময় তাদের কী কী দরকার পড়ে, সেই দরকারগুলোকে আগে থেকে ভাগবাঁটোয়ারা করে রাখা, টেনির হাত থেকে ছিনিয়ে খেতে পারেনা পিলেবান প্যালা, তার জন্যে একটু কিছু তুলে রাখা। ইত্যাদি। তাকে বোঝা যায় সদাসর্বদা, কিন্তু চিহ্নিত করার কাজটা খুব সমস্যার হয়।

এই উপস্থিতিটা আরো প্রকট হয় গ্নু-লিনাক্স সিস্টেমে। প্রতি পদে পদে। এমনকি বিরক্ত হয়ে সিস্টেম বন্ধ করে দিতে চাইলেন, টাইপ করে কমান্ড দিলেন, পাওয়ারঅফ (poweroff), তাতেও ও সঙ্গে সঙ্গে জানাবে, তোমাকে খোকা এই কাজটা করতে গেলে রুট হতে হবে। হার্ডডিস্কের কোনো একটা পার্টিশনে কোনো একটা ফাইল লিখতে বা পড়তে চাইলেন, আপনাকে জানাল, না খোকা, ওটা এখন মাউন্ট করা নেই, মাউন্ট করতে চাইলেন, আপনাকে জানাল, তোমাকে রুট হতে হবে। পার্টিশন কী তা আমরা পরে জানব, পার্টে পার্টে এগোতে হবে না? মাউন্ট ব্যাপারটাও তাই, ইস্টবেঙ্গল মাঠের র‍্যামপার্টের নালার পাশে তাড়া করে আসা দামী চকচকে ঘোড়ার পিঠে শুঁটকো কনিষ্ঠবলের মানে মাউন্টেড পুলিশের হাতের ব্যাটনের চেয়ে অনেক কম জটিল একটা বস্তু। বেশ শিখে যাব আমরা, মানে শিখতে

আমাদের হবেই। রুটের এই যত্রতত্র রুট বন্ধ করে দিয়ে ট্রাফিক থামিয়ে মন্ত্রীর মহামন্ত্রীর নাম নেওয়ার অত্যাচারটা মাঝে মাঝে রীতিমত বেদনার হয়ে ওঠে, গোটা ব্যাপারটা ঠিক করে না-বুঝলে। অজস্র কমান্ড আছে যেগুলো শুধু রুটই চালাতে পারে, আপনি কমান্ডটা টাইপ করে এন্টার মারুন, আপনাকে শুনিয়ে দেবে, কমান্ড খুঁজে পাইনি, নট ফাউন্ড। অথচ সু হয়ে নিন, কমান্ড প্রম্পটে সু (`su` — **S**uper-**U**ser/**S**ubstitue-**U**ser) কমান্ড দিয়ে, দেখবেন সেই একই কমান্ড চমৎকার চলছে। সু হতে গেলে আপনার অবশ্য রুট পাসওয়ার্ড জানতে হবে, কারন সিস্টেম সেটা চাইবে।

এগুলো আসলে গত দিনের একদম শেষে আমাদের বুড়ি ছুঁয়ে চলে আসা নিরাপত্তার গল্পেরই পার্ট, গ্নু-লিনাক্স তথা যে কোনো ইউনিক্স সবসময়েই, আপনা থেকেই, যার টেকনিকাল নাম ডিফল্ট, সেই ডিফল্টেই ধরে নেয় যে, সিস্টেমটা একটা নেটওয়ার্কে আছে। মাল্টি-ইউজার তো বটেই, মাল্টি-মেশিনও। সেখানে একজন সুপারভাইজর বা সুপারইউজার বা রুটের একচ্ছত্র স্বৈরতান্ত্রিক উপস্থিতি অত্যাবশ্যক। কিন্তু আমরা যারা স্ট্যান্ড অ্যালোন মানে একা একলা একটা পিসিতে কাজ করি, তাদের প্রায়ই খাজনার চেয়ে বাজনা বেশির একটা অনুভূতি তৈরি হয় এতে। কিন্তু পরে দেখব আমরা, একটা কদাচিৎ কখনো ডায়াল আপ মোডেম দিয়ে নেটনিষিক্ত, মোটের উপর একা একলা একটা টেবিলে বিরাজ করা একটা পিসিতেও কতটা জরুরি হয়ে ওঠে এই নিরাপত্তার ব্যাপারটা, মূলত ভাইরাস সংক্রমণের হাত থেকে বাঁচতে। আমি নিজে গ্নু-লিনাক্সে এসেছিলাম এই ভাইরাস থেকে বাঁচতে। আমার বাড়িতে একটা স্টাডি সার্কলে আসত তথাগত, পরপর ভাইরাসে আমাকে জেরবার দেখে বলেছিল, তুমি লিনাক্সে চলে এসো দীপঙ্করদা, তখন দুচারটে করে ভাইরাস ফোল্ডারে ফোল্ডারে পুষতেও পারবে, যদি চাও — আমি নিজে তো একটা ভাইরাস সংগ্রহ তৈরি করব ভাবছি — লিনাক্সে ওরা সম্পূর্ণ অকেজো, সেই ভাবেই বানানো অপারেটিং সিস্টেমটা। পরে আমরা দেখব ভাইরাসে এই অসংক্রমণীয়তাটা গোটাটাই এই নিরাপত্তা ব্যবস্থার অংশ।

কিন্তু মুহূর্মুহূ যখন সিস্টেম খুঁতখুঁত করতে থাকে, এটা কোরোনা, ওটা ছুঁয়োনা, ওদিকে তাকিয়ো না, তখন সত্যিই বেশ বিরক্ত লাগে। এর থেকে বাঁচার একটা উপায় হল, সিস্টেমের `su`-জন হওয়ার হদিশ, বাঁশরী-সঙ্কেত, মানে রুট পাসওয়ার্ডটা মনে রেখে দেওয়া, শুধু কুজন মানে ইউজার থেকে প্রায়ই চলেনা, কূজনটুকুও চলেনা প্রায়ই। এর থেকে বাঁচার কিছু উপায় কোনো কোনো গ্নু-লিনাক্স ডিস্ট্রিবিউশন ইনস্টলেশনের সময় দিয়ে থাকে — কী চাও বাছা, এমন ব্যবস্থা করে দেব যে কোনো ব্যবহারকারীকেই সিস্টেম রুট বলে ভাববে? কিন্তু এর মানে, গ্নু-লিনাক্স সিস্টেমের সবচেয়ে জোরালো একটা জায়গাকে আপনি প্রথমেই বিসর্জন দিয়ে এলেন। শুধু আপনি কেন, আক্রমণকারী ভাইরাস যখন চলতে চাইবে, ভাইরাস-রা হল একরকমের প্রোগ্রাম, যারা সঙ্গত ব্যবহারকারীর ইচ্ছে অনিচ্ছের তোয়াক্কা না-রেখেই চলে, এবং সিস্টেম জুড়ে অবাঞ্ছিত খুনখারাপি ঘটাতে থাকে, তখন সেই ভাইরাসকেও সিস্টেম রুট বলে ভাববে, এবং নিজের অন্দরতম অভ্যন্তরও ওয়াইড ওপেন করে দেবে তার সামনে, যেটা শুধু রুটের সঙ্গেই করার কথা তার।

এই পাঠমালার পরের একাধিক দিনে এই নিয়ে আলোচনা তুলব আমরা, গ্নু-লিনাক্স কাঠামোর এই নিরাপত্তা ব্যবস্থা নিয়ে। যার মূল উপাদান হল একটা ফাইল পড়ার বা দেখার বা চালানোর অধিকার, এবং একটা ডিরেক্টরিতে ঢোকার বা সেখানে কী কী ফাইল আছে সেটা দেখার অধিকার, এবং ফাইল বা ডিরেক্টরির উপর মালিকানা। বহু ডিরেক্টরিতে আপনি ঢুকতেই পারবেন না, কারণ সেই ডিরেক্টরিতে ঢোকার অনুমতি নেই আপনার। বহু ফাইল চালানোর, বদলানোর অনুমতি তো ছেড়েই দিন, এমনকি পড়ার অনুমতিই নেই আপনার। সুজন-সঙ্কেত না দিলে, সেই ফাইলে কী আছে ফাইলে সেটাই দেখতে পাবেন না আপনি। নিরাপত্তার খ্যাঁচটা এখানেই — আপনার যেমন নেই, ভাইরাসেরও নেই। অনেকটা নিজের মাত্রা টেনে ভাইরাসের যাত্রাভঙ্গ। কোনো সিস্টেম ফাইলই বদলানো তো দূরের কথা পড়ার অনুমতিই যদি না থাকে তার, তাহলে সে বেচারা আর কী করবে, বসে বসে ভাইরাসান্ডা ভাজা ছাড়া? শুধু তথাগতর না, এখন আমার সিস্টেমেও শত ভাইরাস বিকশিত হোক, শত না হোক নিদেন দু-চারটে ভাইরাস। আপনারা কেউ চাইলেই আনন্দের সঙ্গে দিতে পারি, আমার মন খুবই নরম।

ভাইরাস একটা গ্নু-লিনাক্স সিস্টেমে একমাত্র তবেই সংক্রমণ করতে পারে যদি আপনি রুট হয়েই ভাইরাসটাকে সক্রিয় করেন। তাতেও অবশ্য কিছু জাটিল্য আছে। এই পাঠমালার একদম শেষ দিকে মানে আট নয় আর দশ নম্বর দিনে, যখন আমরা গ্নু-লিনাক্স ফাইলব্যবস্থা আর কনফিগারেশন নিয়ে আলোচনা করব, তখন বুঝবেন। যাইহোক,

ওরকম আত্মপিসিঘাতী হওয়ার বাসনা থাকলে তার সহজতর অনেক পন্থা আছে। আর এরকম ঘাতক হবেন কেন, হিংসা পরম অধর্ম, পরম ঘৃণা এবং তাচ্ছিল্যের সঙ্গে আপনার পিসিটাকে জিএলটিতে দিয়ে দিন, একটা বাড়তি মেশিন রাখতে পারলে বেশ সুবিধে হত। অনেকেরই বেশ সুবিধে হত।

যাকগে, যেখান থেকে শুরু করেছিলাম, অপারেটিং সিস্টেমের মার খুব খাওয়া যায়, কিন্তু ঠিক কোথা থেকে মারছে সেটা খুব স্পষ্ট করে চিহ্নিত করা যায়না। অনেকটা আমাদের এই স্বাধীন তৃতীয় বিশ্বের মত। আগে ব্যাপারটা ছিল স্পষ্ট চিহ্নিত, তারাপদ রায়ের কুকুরের গলার বকলসে যেমন লেখা থাকত, 'আমি পণ্ডিতিয়ার তারাবাবুর কুকুর, আপনি কোন বাবুর কুকুর?', বাবুত্বটা সেখানে নিশ্চিত এবং নির্দিষ্ট, অঙ্কের ভাষায় বললে, ফাইনিট এবং ডেফিনিট। কোনো দেশের ছিল ইংরেজ বাবু, কারোর ফরাসি বাবু, এরকম, এখনও, প্রতিমুহূর্তেই, আঙিনা জুড়ে বাবুর পদচিহ্ন পাই, বাবু তো আছেই, কিন্তু ঠিক কোন বাঁধাবাবু আমায় রেখেছে সেটা বলা শক্ত, রোজই এত লোককে ঘরে বসাতে হচ্ছে, নানা এমএনসির নানা ব্রান্ডের নানা লোক। ঔপনিবেশিক বা উত্তরঔপনিবেশিক, বাবু আমাদের আছেই, শুধু তাকে দেখানো যাচ্ছেনা আলাদা করে। ঠিক সেরকম সব কিছুই তার হাতে, সব হেঁসেলের চাবি, সব শিকের হদিশ, কিন্তু তাকে আলাদা করে টেনে এনে দেখানো খুব শক্ত। এই হেঁয়ালির একটা বড় কারণ এই যে, একদম শুরু থেকেই অপারেটিং সিস্টেম এমন দুটো কাজ একই সঙ্গে করে চলেছে যাদের মধ্যে সরাসরি কোনো সম্পর্ক নেই, মানে একটা কাজ অন্য কাজটার উপর সরাসরি নির্ভরশীল নয়। আরো দুটো কাজই এমন শক্তিশালী এবং প্রবল রকমে তার উপস্থিতি চাগিয়ে রাখে সিস্টেমে, যে একটা কাজের আইডেন্টিটির মধ্যে অন্যটা ঢুকে আসে, মোটের উপর গোটাটাই ঘেঁটে দেয়।

> কাজ নম্বর এক। মেশিনটাকে মেশিনের বাইরে বাড়িয়ে তোলা, পরিবর্ধিত মেশিন করে তোলা। ভৌত মেশিনের সাথে একই সঙ্গে তাকে সাঙ্কেতিক মেশিন ভৌতিক মেশিন করে দেওয়া। দেরিদাদা যেমন বলেছিলেন ইউরোপের বাইরে ইউরোপের বেড়ে চলার কথা। ঠিক ভৌত উপাদানগুলোকে মিলিয়ে মেশিনের যে ভৌত উপস্থিতি, তার বাইরে মেশিনকে বাড়িয়ে তোলা — এক্সটেন্ডিং দি মেশিন।
>
> কাজ নম্বর দুই। মেশিনের এই তুমুল বাড়াবাড়ির আমলে, ভৌত মেশিনকে সাঙ্কেতিক তথা ভৌতিক তথা ভারচুয়াল করে দেওয়ার সঙ্গে সঙ্গে মেশিনের ভৌত উপাদানগুলোর যে সম্মিলিত রসদ বা রিসোর্স — তার প্রয়োজনমাফিক সঠিক ব্যবহারের ব্যবস্থা করা। রসদগুলো যাতে বাজে খরচ না-হয় তার পাহারা রাখা। সিস্টেমের আদত রসদগুলো এলোমেলে হয়ে যেতে না-দেওয়া, ঘেঁটে যেতে না-দেওয়া। রিসোর্স ম্যানেজমেন্ট।

আর ঘটনাটা এই যে, অপারেটিং সিস্টেম নিয়ে কে বলছে এবং কেন বলছে তার উপর দাঁড়িয়ে, কোন অবস্থায় কোন প্রয়োজনে বিচার করছি, তার উপর নির্ভর করে, কখনো এই কাজটা বেশি গুরুত্ব পায়, কখনো ওইটা, কিন্তু দুটো কাজ একসঙ্গে কিছুতেই পিকচারে আসে না। কাজ দুটোকে একটু গুছিয়ে আলোচনা করা যাক।

২।। মেশিন মেশিনতর মেশিনোত্তর — অপারেটিং সিস্টেম মানে পরিবর্ধিত মেশিন

যেমন আগেই বলেছি আমরা, মেশিনের যে ভৌত গঠন বা আর্কিটেকচারটা খুব আদিম জংলি টাইপের, একদম রক্তমাংসের লেভেলে খুব টুকিটাকি কিছু ছাড়া তাকে শেখানো খুব শক্ত। আর্কিটেকচার বলতে এখানে আমরা ঠিক কী কী বুঝি তার একটা তালিকা তৈরি করে ফেলা যাক। এদের উপরেই দাঁড়িয়ে থাকে একটা কার্যরত চালু মেশিনের অপারেটিং সিস্টেম থেকে অ্যাপ্লিকেশন অব্দি হার্ডওয়ার আর সফটওয়ারের গোটা ইমারতটা।

(১) প্রাথমিক আদেশমালা বা ইন্স্ট্রাকশন সেট।

> মেশিনে বিদ্যুৎ অন করার পরই একদম প্রাথমিক চালু হওয়ার বা বুটের সময়, মানে মেশিন যখন ক্রমে ক্রমে কাজ করার উপযুক্ত অবস্থায় আসছে, তখন কী করতে হবে সেটা কম্পিউটারকে বলে দেয় এই ইন্স্ট্রাকশন সেট। এর উদাহরণ বেসিক ইনপুট আউটপুট সিস্টেম বা বায়োস (BIOS — **B**asic-**I**nput-**O**utput-**S**ystem)। কালো স্ক্রিনে যখন লেখা ফুটে উঠছে লাইন লাইন করে সাদা অক্ষরে, দেখবেন একদম নিচের দিকে রয়েছে, সেট-আপে যদি ঢুকতে চাও ডেল মারো — 'Hit DEL to enter Setup'। এই 'সেট-আপ' বায়োস সেট-আপ, বেসিক ইনপুট আউপুট বা তথ্য ঢোকানো বার-করার প্রাথমিকতম কায়দাকানুনগুলো কম্পিউটারকে

বলে দেয় এই বায়োস। বায়োসের বিভিন্ন ব্যবস্থা বা সেটিং-কে বদলানো যায় বায়োস সেট-আপ-এ ঢুকে। সেটা করতে গেলে আপনাকে ওই 'DEL' চাবিটা টিপতে হবে, এবং টিপতে হবে তখুনি, সেই মুহূর্তেই। দেরি করলেই কম্পিউটার সম্পূর্ণ বুট করে যাবে।

এবার ভাবুন, এই যে 'ডিলিট' বা 'ডেল' বা 'DEL' সুইচটা টিকলে এটা করতে হবে, এই কথাটা কম্পিউটার জানছে কী করে? শূন্য নম্বর দিনে করা আলোচনাটা মনে করুন, নিয়ন্ত্রণ তথা সিদ্ধান্ত নেওয়ার অধিকার যার, সেই সিপিইউ জানছে কী করে, যে ওই সুইচটা টিপলে এটা করতে হয়? তাকে কে শেখাল? কী ভাবে শেখাল? অন করলাম আমরা, তার আগে মেশিনটা অফ ছিল, কোনো কাজ করছিল না, সিপিইউতে কিছু নেই, র‍্যামে কিছু নেই, আর হার্ডডিস্ক থেকে পড়তেও শুরু করেনি তখনো, পরে দেখবেন, বুটপ্রক্রিয়াটা আরো কিছুদূর চলার পরে কম্পিউটার হার্ডডিস্ক থেকে পড়তে শুরু করে, তাহলে এটা করতে কম্পিউটার শিখল কী করে? এই শেখানোটা তৈরি করে দিচ্ছে প্রাথমিক আদেশমালা বা ইন্সট্রাকশন সেট। এটা লেখা থাকে রম চিপে, যার কথা আমরা আগেই বলেছি। বলা থাকে, মেশিন অন হল, তোমার মধ্যে বিদ্যুৎ বা প্রাণ এল, তো ইহা ইহা করিতে আরম্ভ করো, শিশুগণ দাও মন নিজ নিজ পাঠে। মেশিনও বুঝে গেল, এখন এই এই লেখা ফুটিয়ে তোলো স্ক্রিনে, যার মধ্যেই পড়ে ওই ডেল টিপলে বায়োস সেট-আপে ঢোকার নেমন্তন্ন। এই প্রাথমিক ইন্সট্রাকশন সেটে যদি থাকত, মেশিন বুট হওয়ার পরেই ঘ্যাঁক করে একটা আওয়াজ করো, তাহলে সে তা-ই করত, তবে সেটা খুবই শক্ত হত, কারণ, ঘ্যাঁক আওয়াজটা একটা জটিল মিডিয়া ফাইল হত, সেই ফাইলকে চালানোর মত প্রোগ্রাম তার আগেই চালু করে দেওয়ার ব্যবস্থা করতে হত।

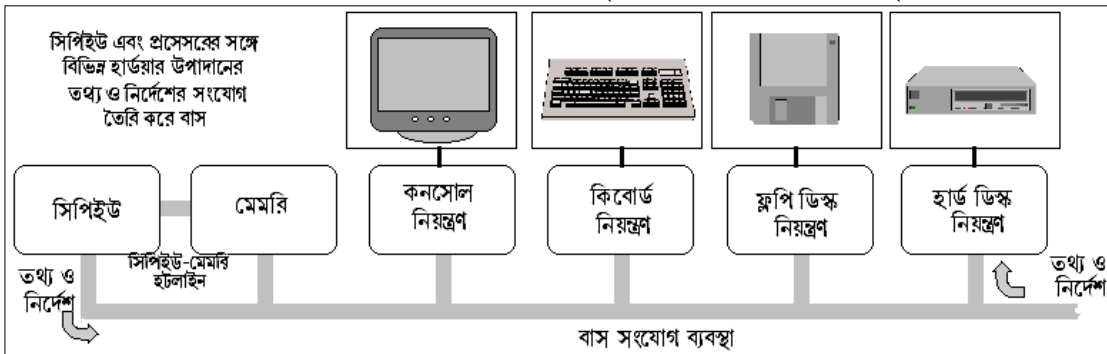
(২) সিস্টেমের নিজস্ব স্মৃতি-সংগঠন বা মেমরি অর্গানাইজেশন।

সিস্টেমের যে স্মৃতি-সংগঠন বা মেমরি অর্গানাইজেশন — সেটাও পড়ে এই আর্কিটেকচার বা ভৌত গঠনের মধ্যে। স্মৃতির কী কী ধরনের ভৌত রসদ ভরা আছে মেশিনে, কী কী পরিমানে, চলমান নানা প্রোগ্রামের মধ্যে তাদের কী ভাবে বণ্টন করবে অপারেটিং সিস্টেম — এই গোটাটা পড়ে এই স্মৃতি-সংগঠন বা মেমরি-অর্গানাইজেশনের মধ্যে। একদম প্রাথমিক আলোচনাটুকু আমরা আগেই করেছি। পরে আরো বহুবার আসবে আলোচনাটা, যখন যখন সিস্টেম বুঝতে গিয়ে দরকার পড়বে।

(৩) কম্পিউটারে তথ্য ঢোকানো এবং বার-করা মানে ইনপুট-আউটপুট বা আইও (I/O)।

আগেই বলেছি, কম্পিউটারের কাজ তথ্য চটকানো। সেই তথ্যকে চটকানোর আগে কম্পিউটারে ঢোকাতে হবে, এবং চটকানো শেষ হওয়ার পরে কম্পিউটার থেকে বার করতে হবে, একেই এক কথায় বলা হয় আইও। শূন্য নম্বর দিনে এই নিয়ে কিছু আলোচনা আমরা করেছি, পরে আরো আসবে। আজকেই, একদম শেষে গিয়ে আরো একটু আসবে।

(৪) কম্পিউটারের বাস-গঠন।

এই বাস বা BUS বলতে বোঝায় একটা কম্পিউটারের বিভিন্ন কেজো ইউনিটগুলোর ভিতর পারস্পরিক যোগাযোগটা যে পথ বেয়ে ঘটে — একটা বিদ্যুৎখচিত পথরেখা — ঠিক আমাদের প্রতিদিনের যাতায়াতের বাসের মতই, তথ্যকে যে কোনো জায়গা থেকে যে কোনো জায়গায় নিয়ে যায়। এই বাসের মধ্যে আবার নানা প্রকারভেদ হয়। নাম থেকেই আন্দাজ করা যায়, লোকাল বাস গুলো সিপিইউ বা সেন্ট্রাল প্রসেসিং ইউনিটের বিভিন্ন অংশগুলোর ভিতর যোগাযোগ রাখে — লোকাল দূরত্বে মানে কাছেপিঠে। আর দূর পাল্লার এক্সটার্নাল

বাসেরা যোগাযোগ রাখে মেমরি এবং আনুষঙ্গিক যন্ত্রপাতি বা পেরিফেরালের সঙ্গে। আবার ঠিকানা জেনে রাখার এবং যোগাযোগের কাজ করে অ্যাড্রেস বাস। অ্যাড্রেস বাসরা খতিয়ান রাখে মেমরির বিভিন্ন এলাকার, কোথাকার তথ্য কোথায় গড়ায়, কোথায় কে যেতে পারে, বা যায়, বা আগে গিয়েছিল — তাই এখন আছে — মাঝরাত্তিরে বিনা ওয়ারেন্টে তুলে আনা যাবে, আরো যদি কলেজ-ইউনিভার্সিটির আবোধা মাস্টার হয়। এই তিন রকম, লোকাল বাস, এক্সটার্নাল বাস আর অ্যাড্রেস বাসদের নিয়ে তৈরি বাস-ব্যবস্থা, এই নিয়ে আরো আলোচনা আছে আমাদের আজই।

উপরের এই চার ধরনের উপাদান, ইন্সট্রাকশন-সেট, মেমরি-সংগঠন, ইনপুট-আউটপুট, আর বাস-গঠন — এদের নিয়ে তৈরি মেশিনের আদিম জংলি আর্কিটেকচারের স্তর। এই স্তরে সে এতটাই জংলি যে এখানে কম্পুবাবুকে কিছু শেখানো কদাকার রকমের শক্ত, বিশেষ করে যদি ঢোকানো-বার-করা বা ইনপুট-আউটপুট হয়।

আগের দিন মানে এক নম্বর দিনে, আমাদের সি-প্রোগ্রাম লেখা এবং রান করানোর কথাটা মনে করুন। সিকিতম সি না জেনেই, শুধু এই কম্পিউটারকে গাধা বলে ডাকার বিমল বিলাসে একটা রীতিমত দীর্ঘর্য শেষহীন নিরবচ্ছিন্ন রকমের লম্বা চারলাইনের gadha.c কোডফাইল লিখে, সেটাকে কম্পাইল করে প্রোগ্রাম বানিয়ে, সেই প্রোগ্রাম চালিয়ে আমাদের C-তল মানে আমেরিকান রকমের 'কুল' C-শিক্ষার প্রমাণ দিয়েছিলাম।

এবার ধরে নিন, ওই কোডটা লিখে, কম্পাইল করে প্রোগ্রাম বানিয়ে, তাকে চালিয়ে যা যা করেছিলাম আমরা, মানে 'কিরে গাধা!!!' বলে ডাকা — সেই একই কাজ করতে হবে আমাদের, কোডে লেখা আদেশগুলোই পালন করতে হবে, শুধু এখন আর কোনো অপারেটিং সিস্টেম নেই, কম্পাইলার নেই, কিচ্ছু নেই। শুধু আদিম এই আর্কিটেকচার বুকে নিয়ে কম্পিউটারটা দাঁড়িয়ে আছে। এই অবস্থায় দাঁড়িয়ে, ঠিক 'gadha.c' কোডের থেকে তৈরি 'gadha' প্রোগ্রামের কাজটাই করতে হবে আমাদের। মানে, ওই উপরে নিচে এক লাইন ফাঁকা রেখে ওই সম্মান সম্বোধনটাকে ফুটিয়ে তুলতে হবে স্ক্রিনে।

এবার ভাবুন, কাজটা শুরু হবে কোত্থেকে? প্রথমে ওই কোডফাইলের আদেশগুলো তাকে পড়াতে হবে, তারপর সেই অনুযায়ী তাকে কাজ করতে বলতে হবে। তার মানে মোট যত জিনিষ আপনার কম্পিউটারকে দিয়ে করাতে হবে, তার মধ্যে কী কী পড়বে তার কিছুটা ভাবার চেষ্টা করা যাক। সিস্টেমকে আমাদের নিঁখুত ভাবে বলে দিতে হবে, যে ডিস্ক থেকে আদেশগুলো পড়বে, সেই ডিস্ক মানে হচ্ছে এই সার্কিটের এই অংশ থেকে এই কানেক্টরের এই সংযোগ। সেখানে যাও। যে কৌটোটা পাবে ভিতরে মোটরটাকে অন করো, মানে এত ভোল্টের এতটা বিদ্যুৎ পাঠাও, ঘোরাও। এবার ঘুরতে থাকা ডিস্কের উপর কাঁটাটা নড়াও। এত এত নম্বর সেক্টর পড়ো। এটা করতে গিয়ে কোন ডিস্কের কোন সেক্টরে কোন কাঁটা কত ডিগ্রি কোণে ঘোরাবে তাও বলে দিতে হবে। তারপর বলে দিতে হবে, ওই আদেশমালাকে মেমরির এত এত নম্বর সেক্টরে এইভাবে তোলো। এই ভাবে বলতেই থাকো বলতেই থাকো, এতো সবে কলির সন্ধে। পুরো প্রক্রিয়াটা শেষ হবে স্ক্রিনে, মানে সার্কিটের এই অংশের এই সংযোগে এই এই পিক্সেল বা আলোকবিন্দু উদ্দীপ্ত করবার জন্যে তুমি এই এই পরিমাণ বিদ্যুৎ পাঠাও। শিউরে উঠবেন না, প্রায় এই ভাবেই কাজ করতে হত কম্পিউটারে, মাত্র দশক চারেক আগেই।

এবার দ্বিতীয় স্টেপ ভাবুন। আর একটা সিচুয়েশন। এবার কাজটা আর অতটা বীভৎস নয়। ধরুন কম্পাইলারটা আছে আমাদের কাছে। মানে, আমাদের সি-তে লেখা কোড বা আদেশমালাকে সে অনুবাদ করে দেবে কম্পিউটারবোধ্য মেশিনভাষায়, আমাদের ওই আদিম আর্কিটেকচার যা বোঝে। শুধু ইনপুট-আউটপুটের ডে খুঁটিনাটিগুলো আপনার হয়ে অপারেটিং সিস্টেম নিজেই বুঝে নেয়, সেটা সে আর বুঝছে না। তাই ইনপুট-আউটপুট বা আই-ও সংক্রান্ত ডিটেইলস আমাদের নিজেদেরই জানিয়ে দিতে হবে। মানে, এবারেও কাজটা ভাবুন, অপারেটিং সিস্টেমকে আপনার নিজেরই জানিয়ে দিতে হবে, যে ডিস্কে লেখা আছে প্রোগ্রামটা সেই ডিস্ক মানে ঠিক কোন ভৌত উপাদানটা, এবং সেখান থেকে তথ্য পড়া মানে ঠিক কী করা বোঝায় সেটাও বলে দিতে হবে আপনাকে। বলে দিতে হবে, স্ক্রিনে লেখা মানে কী, এবং স্ক্রিনের ভৌত উপাদানের খুঁটিনাটিও বলে দিতে হবে একই সঙ্গে। কম্পিউটারকে গাধা বলে ডাকার বাসনাটা যতই গাঢ় হোক, আপনার আর ইচ্ছে হচ্ছে প্রোগ্রামটা চালানোর?

অপারেটিং সিস্টেমের কাজ ঠিক এটাই। আপনার হয়ে অনেকটা কাজ করে রেখে দেওয়া। বিভিন্ন ভৌত উপাদানকে, তাদের অংশগুলোকে চিনে রাখা, তাদের নাম দিয়ে রাখা। তাদের কাজ করার খুঁটিনাটি নিজের মধ্যে ভরে রাখা, যাতে আপনার কাজের প্রয়োজন হওয়া মাত্রই, আপনি ঠিক যে ভাবে চান সেভাবেই পেতে পারেন সিস্টেমটাকে।

প্রোগ্রামচিদের চাই একটা সরল সহজ সোজাসাপ্টা ভূমি, তার নাম যাই হোক, '/home/user' বা '/mnt/floppy', কিম্বা উইন্ডোজে 'C:\' বা 'A:\'। ওই নামগুলোর পিছনে ভৌত জায়গাগুলোকে আলাদা করে চিনিয়ে দিতে হবেনা। সেখানে 'খোলো' বা 'open' বললেই ফাইল খুলে যাবে, কিবোর্ডে টাইপ করলেই নতুন ঝকঝকে সাইবারপ্রোজ্বল সুখিনী বর্ণমালাকে যোগ বিয়োগ করা যাবে, 'তুলে রাখো' বা 'save' বললেই সঞ্চয় হয়ে যাবে ফাইলটা। খোলা বা তুলে-রাখা মানে কী সেটা সিস্টেম নিজে নিজেই বুঝে যাবে। এমনকি টাকা-পয়সা রাখার মত ব্যাংক যাওয়ার ঝঞ্ঝাটটুকু অব্দি করতে হবেনা। অর্থাৎ মেশিনের ওই আদিম জংলি আর্কিটেকচারের প্রাকৃত জগত থেকে অনেক উন্নত এলিট আলোকপ্রাপ্ত একটা সাংকেতিক ভূমি — '/dev/hda1' বা 'C:\' নামে যাকে ডাকা বোঝা এবং বদলানো যাবে। গ্নু-লিনাক্সে বা উইনডোজে আমরা যেভাবে সচরাচর আমাদের হার্ডডিস্ককে বুঝিয়ে থাকি। এই ডাকাটা এখন সাংকেতিক এবং বিমূর্ত, সিম্বলিক এবং অ্যাবস্ট্রাক্ট।

এই ডিস্ক মানে এবার কিছু নির্দিষ্ট ডিরেক্টরি, এবং তাদের পেটের মধ্যে আবার আরো ডিরেক্টরি বা সাব-ডিরেক্টরি। আর ডিরেক্টরিগুলোকে ফাঁকা রাখার জন্যে তো আর আপনি অত পয়সা খরচ করে মেশিন কেনেননি। ডিরেক্টরি থাকলেই তাদের ভরে দিতে হয় ফাইল দিয়ে। তাই ডিরেক্টরির পর ডিরেক্টরি ভর্তি গুচ্ছ গুচ্ছ ফাইল। যার প্রতিটি ফাইলই আপনি চাইলে খুলতে পারেন, বদলাতে পারেন, বন্ধ করতে পারেন, তাদের নিয়ে কাজ করতে পারেন, আপনার নিজের পয়সায় কেনা মেশিন বলে কথা। এবং এতো অনামিকাকে অনাম্নীর লেখা চিঠি নয় যে 'আনজান লিখ্খা থা উপর, নিচে নেম ছে', অর্থাৎ তাদের প্রত্যেকটি ফাইলেরই একটা নাম আছে, যা দেওয়া যায়, বদলানো যায়, অন্য নামে একই ফাইলের প্রতিরূপ বানানো যায় বা কপি করা যায়। এখন থেকে আপনি শুধু ফাইলের নাম, ডিরেক্টরির নাম, আর ড্রাইভের নাম জানলেই যে কোনো ফাইলকে নিয়ে যে কোনো কাজ করতে পারবেন।

এই সাংকেতিক নাম দিয়ে নির্দিষ্ট ভূমিটা কিন্তু আর ভৌত মেশিন নয়। মেশিন মানে ছিল কিছু বৈদ্যুতিক তার, কিছু সিলিকন চিপ, কিছু ধাতুর চাকতি, কিছু ডান্ডা, কিছু পেরেক, কিছু ইত্যাদি — সেই বদখত জটিল এলোমেলো চেহারাটাকে এমন নামময় করার রূপটান দেওয়াটাই অপারেটিং সিস্টেমের কাজ। শুধু সফট বা হার্ড ডিস্কের কঠিন চাকতিময় খুঁটিনাটি নয়, আমাদের ভুলতে দেয় এদের ইন্টারাপ্টগুলোকেও।

ইন্টারাপ্ট মানে কিছু সিগনাল যা বিভিন্ন ভৌত উপাদানগুলো থেকে অপারেটিং সিস্টেমের কাছে আসে, যেমন সিডি-ড্রাইভ, মাউস ইত্যাদি। অপারেটিং সিস্টেম এই ভৌত উপাদানগুলোকে মাথায় রাখে এই সিগনালদের দিয়ে। সিগনালগুলো আসে একটা ডিভাইস ড্রাইভার বা উপাদান চালক মারফত। ডিভাইস ড্রাইভার মানে বিশেষ রকমের কিছু আদেশ-সমাহার। যা দিয়ে অপারেটিং সিস্টেম একটা উপাদানকে ব্যবহার করে। ধরুন আপনি হার্ড-ডিস্ককে কাজে লাগানোর সময় যে সব আদেশ দিয়ে অপারেটিং কাজ করতে পারবে, মাউস বা সিডি বা স্ক্যানার ব্যবহার করার সময় সেই আদেশগুলো দিয়ে চলবে না। সমাহারটা বদলে যাবে। মাউসের জন্যে লাগে মাউস ড্রাইভার। আপনার মনিটরটাকে ব্যবহার করার জন্যে লাগে মনিটর ড্রাইভার। এই রকম আছে প্রত্যেকটা উপাদানেরই।

শুধু হার্ডওয়ার বা ডিভাইসদের ইন্টারাপ্টগুলোকে ভুলে থাকতে দেওয়া নয়, এই অপারেটিং সিস্টেমের কল্যাণেই আমরা ভুলে থাকতে পারি টাইমার-গুলোকে, কম্পিউটারের গভীর জটিল রহস্যময় উদরে এবং অন্ত্রে নিয়ত খেলাধূলামান ক্লক সাইকেলের ওঠানামাকে। ভুলে থাকতে পারি কতটা মেমরি বা স্মৃতিকে কী ভাবে ব্যবহার করা হচ্ছে — প্রতিমুহূর্তে — তার জটিল ছকটাকে।

ধরুন আপনি একটা কবিতা লিখছেন ওয়ার্ড-প্রসেসরে, তার ফাইলটা আছে ফ্লপিতে, আপনি যে বদলগুলো আনছেন সেগুলো লিখে রাখতে বলেছেন ফ্লপি নামক ভাঁড়ারে অমুক নামের ফাইলে। একটু আগে লেখা এবং হার্ড ডিস্কে সেভ করে রাখা আপনার যুগান্তকারী প্রোগ্রামগুলো কম্পাইল হচ্ছে অন্য আর এক দিকে। কবিতা লেখার প্রহরটাকে যে নান্দনিক উত্তাপে আপনি ভরে তুলছেন, সেই যে-নামেই-ডাকো-গোলাপের ছবিগুলো আছে একটা গ্রাফিক্স-

সিডিতে। সেখান থেকে একটা একটা করে গোলাপ তুলে এনে স্ক্রিনের ভারচুয়াল ফুলদানিতে ভরে দিচ্ছে একটা গ্রাফিক্স সফটওয়ার।

এবার, নতুন লাইনে তা দিতে গিয়ে আপনি সেই ছবির ফুল থেকে ফুলান্তরে গেলেন। আপনি তো কমান্ড দিয়ে বা মাউস টিপেই খালাস, কম্পিউটারকে তখন কী কী করে যেতে হচ্ছে নির্বাক বশংবদতায়, সেটা মাথায় এনেছেন কখনো? নিমেষে এই প্রতিটি নামের ঠিকানায় প্রতিটি চালু থাকা কাজের প্রতিটি কার্যরত ফাইলের সমস্ত একটা বিরাট অংশ তথ্যকে আপাতত অস্থায়ী একটা রকমে হার্ডডিস্কের সোয়াপ-ফাইল নামে একটা বিশেষ জায়গায় লিখে রেখে জ্যান্ত র‍্যামে তুলে আনতে হল ওই নতুন ফুলের নতুন কোটি কোটি পিক্সেলের নতুন অর্বুদ অর্বুদ বাইট সমগ্রকে — এই গোটা কাজটাই হয়ে গেল আপনার অগোচরে (হার্ড ডিস্কে অস্থায়ী রকমে লিখনীয় এই সোয়াপ ফাইল নামের জায়গাটার কথা পরে বারবার করে জানতে হবে আমাদের)। আপনি শুধু নাম ধরে ডেকেই খালাস, কাজের নাম ধরে কাজটা করার কথা জানিয়েই খালাস। আপনার হয়ে এই গোটাটাই মাথায় রেখেছে অপারেটিং সিস্টেম। আপনি যখন অমুক পার্টিশনের অমুক ডাইরেক্টরির অমুখ ফাইলে অমুক কাজ করতে বলছেন, এই প্রত্যেকটা অমুকই ভৌত নয় সাঙ্কেতিক, সে বুঝে নিচ্ছে সেই সাঙ্কেতিক অমুক মানে কোন ভৌত তমুক, জানে এবং হিশেব করে নিচ্ছে, নিঃশব্দে। এইভাবেই আপনার মেশিনকে মেশিনতর করে মেশিনোত্তরতায় ঠেলে দিচ্ছে, হার্ডওয়ার থেকে সফটওয়ারের জগতে টেনে আনছে সাঙ্কেতিক ভূমি মারফত — অপারেটিং সিস্টেম নামের ওই নির্বাক দাস।

৩।। ভৌত উপাদানদের নিয়ে তৈরি সাম্রাজ্য

কিন্তু অপারেটিং সিস্টেমকে ব্যবহারকারীর কাছে এই রকম একটা সহজ পারস্পরিকতা বা কনভিনিয়েন্ট ইন্টারফেস হিশেবে দেখাতে গিয়ে আমরা গোটা ব্যাপারটাকে উপর থেকে দেখছি — ব্যবহারকারীরা যেখানে রয়েছে — একটা মেশিনের হাই লেভেল বা উঁচু স্তর থেকে। ইউজার যেখানে প্রয়োগ করে চলেছে হাজার একটা অ্যাপ্লিকেশন। উল্টো একটা দৃষ্টিকোণ থেকেও পুরোটাকে দেখা যেতে পারত — হার্ডওয়ারের দিক থেকে, নিচুর দিকে থেকে, মানে গোটা কম্পিউটার সিস্টেমটাকে তার মাথার উপরে শীর্ষাসন করিয়ে। যখন আমাদের সামনে রয়েছে ওই ভৌত উপাদানের রসদগুলো তাদের ভৌত বাস্তবতা নিয়ে, তাদের ক্ষমতা এবং ক্ষমতার সীমা নিয়ে। এটাই কম্পিউটারের রসদের ভাঁড়ার, যাকে ব্যবহার করে কাজ করে চলতে পারে ওই অ্যাপ্লিকেশনগুলো। এই বিভিন্নমুখী বিচিত্র উপাদানের রসদগুলোকে সামাল দেওয়ার কাজটাও অপারেটিং সিস্টেমের। এর পরের সেকশনে আমরা ফেরত আসব রসদের ভাঁড়ারঘরের ভাণ্ডারীর ভূমিকায় অপারেটিং সিস্টেমের আলোচনায়। তার আগে এই রসদগুলোকে একটু দেখে নেওয়া যাক।

সচরাচর আমরা যে পিসিগুলোকে দেখি তাদের ক্ষেত্রে এই ভাঁড়ারের সবচেয়ে জনপ্রিয় উপাদানগুলো নিয়ে আমরা শূন্য নম্বর দিনে আলোচনা করেছি। কিন্তু সেখানে আমরা একটা পিসিকে বাইরে থেকে, একটা ব্যবহারকারীর চোখ থেকে দেখছিলাম। এখানে আমরা দেখছি এদের এক একটা রসদ হিশেবে, কার্যকারিতার আকর হিশেবে, অপারেটিং সিস্টেম তাদের এক এক জনকে দিয়ে এক একটা কাজ করিয়ে নেয়। অপারেটিং সিস্টেমের অন্তর্গত অংশ হিশেবে এদের এবার ভাবছি আমরা। এদেরকে ভিতরে নিয়েই তৈরি হয় একটা অপারেটিং সিস্টেম। অপারেটিং সিস্টেম — এই ব্যাপারটা একটা বিমূর্ত ধারণা, সেটা আকার পায়, মূর্ত হয়ে ওঠে এই উপাদানগুলোর শরীরেই। কাজ করতে থাকা সিপিইউটা একটু অপারেটিং সিস্টেম, হার্ডডিস্কটাও তাই, মেমরিও তাই। একটু আগে দেওয়া বাস-সংযোগের ছবিটা একবার দেখে নিন, সিপিইউ, মেমরি, আই-ও ডিভাইস, এরা সবাই একটা সিস্টেম বাস দিয়ে সংযুক্ত থাকে পরস্পরের সঙ্গে, তৈরি হয় একটা সমগ্র, যে সমগ্রটা হল অপারেটিং সিস্টেমের সাম্রাজ্য। শর্তহীন সম্রাট সে। আর সবাই সেখানে ছোট ছোট খণ্ডের মনসবদার জায়গিরদার, সম্রাটের সিদ্ধান্তের অধীন। আধুনিক একটা পিসিতে বাসের কাঠামো অবশ্য ওর চেয়ে অনেক জটিল হয়। একাধিক বাসের একটা সমাহার কাজ করে সেখানে। তবে সেসব জটিলতায় আমরা যাব না আমাদের এই পাঠমালায়।

> মাদারবোর্ড — নানা কোম্পানির নানা ক্ষমতার মাদারবোর্ড হয়, নানা ধরনের। এক একটা মাদারবোর্ডে এক এক ধরণের ভৌত উপাদান লাগানো যায়, এক এক মাত্রার এক এক মানের কাজ হয়। তাদের দামও হয় আলাদা আলাদা। কম্পিউটারের অন্য উপাদানগুলো কী ক্ষমতার কী মানের হবে তা মোটামুটি ভাবে নির্দিষ্ট

হয়ে যায় এই মাদারবোর্ড দিয়েই। মাদারবোর্ড গোটা ব্যাপারটাকে কোল পেতে দেয়। তার গায়েই লাগানো থাকে প্রসেসর, র‍্যাম, রম, আইও ডিভাইস। তার গায়ের সার্কিটগুলো দিয়েই কাজ করে বাস সংযোগ। বিদ্যুৎ সরবরাহকে প্রতিটি একক ভৌত উপাদানের কাছে পৌঁছে দেয় মাদারবোর্ড। কোনো একটা পিসিতে ঠিক কোন মানের কোন ক্ষমতার কী কী উপাদান লাগানো যাবে, প্রসেসর থেকে শুরু করে প্রতিটি কিছু, স্থির হয়ে যায় মাদারবোর্ড দিয়ে। সেই মাদারবোর্ডে কোন ধরনের কোন উপাদান ব্যবহার করা যাবে, এবং কোন বা কোন কোন উপাদান ব্যবহার করা যাবেনা, তার নির্দিষ্ট সব নিয়ম আছে। এই মাদারবোর্ডের ভৌত ব্যাকরণ তাই অনেকটাই স্থির করে দেয় অপারেটিং সিস্টেমের কাজের প্রকরণকে।

প্রসেসর — এই প্রসেসরই হল একটা কম্পিউটারের মস্তিষ্ক, আগেই বলেছি। বহু কোম্পানির বহু রকমের বহু গঠনের হতে পারে, গতি বা মেগাহার্জ হতে পারে অজস্র রকমের, কাজ করার একক মানে ওয়ার্ড হতে পারে ১৬ বিট থেকে ৬৪ বিট বা তারো বেশি এই নানা সাইজের, সবচেয়ে জনপ্রিয় ব্র্যান্ডগুলো অবশ্যই পেন্টিয়াম অ্যাথলন সাইরিক্স ইত্যাদি। এই প্রসেসরের কাজ হল মেমরি বা স্মৃতি থেকে নির্দেশ বা নির্দেশাবলী তুলে আনা এবং তাদের পালন করা। কোন ধরনের নির্দেশ কোন সিপিইউ পালন করতে পারবে তার খুব নির্দিষ্ট কিছু নিয়ম, এই রকম দু-একটা নিয়ম আমরা পরে উল্লেখ করব। সিপিইউ-র কাজ করার রকম, তার অংশগুলো, এবং তার রেজিস্টারগুলো সম্পর্কে একটা আবছা ধারণা আমাদের ইতিমধ্যেই হয়েছে। এর পরে এর কোনো কোনোটার সফটওয়ার অংশটা নিয়ে দু-চারটে কথা বলব আমরা। তবে এর চেয়ে বেশি কিছু আমাদের এই পাঠমালায় আসবে না। এটা এত বেশি টেকনিকাল এলাকা যে সেখানে কোনো কথা বলার যোগ্যতাই নেই আমার।

মেমরি — কম্পিউটারের ভৌত উপাদানগুলোর মধ্যে সিপিইউর পরেই গুরুত্বপূর্ণতম এই মেমরি। এর নানা ধরনের প্রকারভেদের কথা আমরা আগেই বলেছি, শূন্য নম্বর দিনে। র‍্যাম আর রম — এই দুইরকম স্মৃতি আমাদের সবচেয়ে বেশি পরিচিত। এর পরে আসে ক্যাশে, যা নিয়ে পরে আমরা অনেক খুঁটিয়ে আলোচনা করব। এর পরে আসে নানা ধরনের স্মৃতিভাঁড়ার, মানে মূলত ম্যাগনেটিক ডিস্ক মানে হার্ডডিস্ক বা ফ্লপিডিস্ক, আর অপটিকাল ডিস্ক মানে সিডি ডিভিডি ইত্যাদি। এদের নিয়ে বহু আলোচনা আমাদের এই পাঠমালায় বারংবার আসবে। আসবে হার্ডডিস্ক এবং সিডির আভ্যন্তরীন কাঠামো এবং সেখানের ফাইলব্যবস্থা নিয়ে দীর্ঘ আলোচনা। সিপিইউ রেজিস্টার, ক্যাশে, এবং র‍্যাম নিয়ে খুব ভালো করে জানব আমরা ভারচুয়াল মেমরি বা ভৌতিক স্মৃতির আলোচনায়, আট নম্বর দিনে গিয়ে। মেমরি এবং স্টোরেজ ডিভাইস — সবরকম স্মৃতিকেই মাপা হয় বাইট কিলোবাইট মেগাবাইট গিগাবাইট টেরাবাইটের মাপে, যাদের সম্পর্কেও পরে অনেক ভালো করে জানতে হবে আমাদের। হার্ডডিস্কের বেলায় আর একটা বড় জিনিষ হয়ে দাঁড়ায় তার আর-পি-এম বা রোটেশন পার মিনিট। মানে হার্ডডিস্কের মধ্যের প্ল্যাটারগুলো মিনিটে কতবার ঘোরে। ডিস্কটা কতটা দ্রুত কাজ করতে পারে সেটা এর উপরেই নির্ভর করে। প্রতিটি মেমরি উপাদানেরই অজস্র আলাদা আলাদা রকম আছে, যা আলোচনা করার জায়গা এটা নয়।

টাইমার — মাদার বোর্ডের যে সার্কিট্রি সময়গতি খেয়াল রাখে — তারিখ থেকে তারিখে এবং ক্রিয়া থেকে ক্রিয়ায়, মানে কোনো একটা কাজ কতক্ষণ ধরে চলছে — ইত্যাদি। এটা একটা গুরুত্বপূর্ণ উপাদান, অপারেটিং সিস্টেমের অনেকগুলো কাজ ব্যাপকভাবে নির্ভরশীল থাকে এই টাইমারের উপর। শুধু আমরা জানতে চাইলে সময় বলে দেওয়াটা নয়।

কিবোর্ড — নানা লে-আউটের, নানা গুণমানের, নানা রকম চাবি-সমাহারের। এ নিয়ে যথেষ্ট আলোচনা হয়েছে। এর পরে যেটা দরকার সেটা হল সেই কিবোর্ডটা প্রাকটিশ করা, গ্নু-লিনাক্স অপারেটিং সিস্টেমে কাজ করতে হলে টাইপটা আপনাকে এত প্রচুর করতে হবে, যে এটা খুব প্রয়োজনীয় ইনভেস্টমেন্ট। গ্নু-র তৈরি সফটওয়ার আছে, জিটাইপিস্ট, বেজায় চমৎকার জিনিষ।

পয়েন্টিং ডিভাইস — মানে মাউস (সিরিয়াল না পিএসটু না ইউএসবি বা হুইল বা অপ্টিকাল ইত্যাদি), বা ট্র্যাকবল বা এই জাতের অন্য কিছু, এটাতে কিছু আর জানানোর প্রয়োজন তো নেইই, বরং জানাটা একটু কমিয়ে ফেলতে পারলে ভালো বই মন্দ হবেনা, আরো যেহেতু আমরা আমাদের এই পাঠমালায় গুই বা গ্রাফিকাল-ইউজার-ইন্টারফেস যেহেতু ধরবই না।

কনসোল বা ভিডিও মনিটর — মনো প্রায় উঠেই গেছে এখন, কালার মনিটর অজস্র রকমের আয়তন এবং ক্ষমতার, তাদের পরস্পরের মধ্যে নাটকীয় তফাত, এই ক্ষমতার পার্থক্যকে আবার ব্যবহার করার মত সুযোগ থাকা চাই মাদারবোর্ডে সংযুক্ত ভিডিওকার্ড বা ভিডিও-কন্ট্রোলারের, যার মাধ্যমে কম্পিউটার কনসোলকে ভিডিও তথ্য পাঠায়। এক্স-উইনডোজ মানে গুই যেহেতু আমরা ছুঁচ্ছি না, তাই মনিটরের জটিলতা নিয়ে মাথা খারাপ করার মিনিমাম প্রয়োজন পড়বে না আমাদের।

প্রিন্টার — লেজার না ডেস্কজেট না ডটম্যাট্রিক্স — কত তার রেজলিউশন কটা পাতা প্রতিমিনিটে ছাপে ইত্যাদি। এটাও অপারেটিং সিস্টেমের একটা গুরুত্বপূর্ণ উপাদান।

নেটওয়ার্ক কলকব্জা — মোডেম বা আইএসডিএন বা ল্যান বা ওয়ান বা কত তার তথ্য স্থানান্তরের হার — কেবিপিএস বা এমবিপিএস ইত্যাদি। অন্য অনেক অপারেটিং সিস্টেমের চেয়ে গ্নু-লিনাক্স অপারেটিং সিস্টেমে এই নেটওয়ার্ক কলকব্জাটা আরো বেশি গুরুত্বপূর্ণ হয়ে ওঠে, তার কারণ তো আগেই বলেছি, গ্নু-লিনাক্স তথা যে কোনো ইউনিক্স সিস্টেমই প্রথমেই মেশিনটাকে নেটওয়ার্কিত বলে ধরে নেয়। হার্ডওয়ার নিয়ে তেমন কথা না-এলেও সফটওয়ার কনফিগারেশন নিয়ে কিছু আলোচনা আসবে আমাদের। নয় নম্বর দিনে গিয়ে।

এবং এগুলো ছাড়াও আরো অজস্র প্রচুর রকমের উপাদান এবং তাদের নিজস্ব নিজস্ব জটিলতা। এবং প্রায় প্রতিমুহূর্তে এদের নতুন নতুন ধরন বাজারে আসছে নতুন নতুন প্রকৌশল নিয়ে, নতুন নতুন কাজের প্রয়োজন মেটাতে। তারা সঙ্গে সঙ্গে তাদের ড্রাইভার সহ ঢুকে আসছে কারনেলে, অপারেটিং সিস্টেমে। এই ড্রাইভার ব্যাপারটা নিয়ে আলোচনা আজই হবে।

বারবার আমরা নানা কাজে বিট বাইট বা কিলোবাইট (কেবি) মেগাবাইট (এমবি) গিগাবাইট (জিবি) — এগুলো ব্যবহার করছি। বিট বা বাইট কাকে বলে আমরা জানি। এর পরে এককগুলো আমাদের অভ্যস্ত দশের গুণিতক থেকে একটু আলাদা। আমরা মেট্রিক সিস্টেমে অভ্যস্ত হই এক কিলো মানে এক হাজার। এখানে আসে হাজারের সবচেয়ে কাছের ২-এর গুণিতক। মানে ১০২৪, মানে ২ এর ১০-তম পাওয়ার। ১০২৪ বিটে এক কিলোবাইট বা কেবি। ১০২৪ কিলোবাইটে এক মেমাবাইট বা এমবি। ১০২৪ মেগাবাইটে এক গিগাবাইট বা জিবি। এর উপরে আছে আরো বড় বড় একক — টেরাবাইট পেটাবাইট এক্সাবাইট জিটাবাইট ইয়োটাবাইট ইত্যাদি। তালিকাসহ এদের নিয়ে বড় করে আলোচনা আসবে আট নম্বর দিনে। আপাতত এটুকুতেই কাজ চলে যাবে। আবার, মোডেম বা আইএসডিএন জাতীয় নেটওয়ার্ক উপাদানগুলোকে আমরা মাপি কত বাইট সে এক সেকেন্ডে স্থানান্তর করতে পারে তার নিরিখে। কেবিপিএস মানে কিলোবাইট-পার-সেকেন্ড। এমবিপিএস মানে মেগাবাইট-পার-সেকেন্ড। একটা কম্পিউটার প্রসেসর কত তাড়াতাড়ি কতটা কাজ করতে পারে তাকে মাপি একই ভাবে কিলো-হার্জ মেগা-হার্জ বা গিগা-হার্জ দিয়ে। এখানেও কিলো মানে ১০২৪। ১০২৪ কিলোতে এক মেগা, ১০২৪ মেগাতে এক গিগা। যাকগে, এবার ফিরে আসা যাক রসদের এই ভাঁড়ারঘরের ম্যানেজার হিশেবে অপারেটিং সিস্টেমের কথায়।

৪।। অপারেটিং সিস্টেম মানে রসদের ভাঁড়ারঘরের দায়িত্ব

এই ভাঁড়ারঘরের ম্যানেজার হিশেবে অপারেটিং সিস্টেমের কাজ হল এই নানামুখী নানাধাঁচের নানা রসদকে নিয়ন্ত্রিত এবং সংগঠিত করে হাজির করা বিভিন্ন সফটওয়ারের কাছে। আগের সেকশনের উদাহরণটাই ধরুন। একটা ওয়ার্ড-প্রসেসর, একটা কম্পাইলার, এবং একটা গ্রাফিক্স সফটওয়ার। আর এদের কাজ করার জন্যে নিচের স্তরে অনুচ্চারিত রকমে আরো অনেকগুলো সফটওয়ার, যে সিস্টেম সফটওয়ার গুলো একটু আধটু পরে আমাদের কাজে আসবে। এর মধ্যে একটা মূল রকম হল যখ বা ডিমন। যারা সিস্টেমের মধ্যে বসে থাকে এবং দিবারাত্র তার নিজের কাজ, মানে সিস্টেমে কী কী ঘটছে তার পাহারা দিয়ে চলে। পাহারা দেয় আর খাপ পেতে অপেক্ষা করে, ঠিক কখন সেই ঘটনাটা ঘটবে যখন তার কিছু একটা করতে হবে। কিছু একটা করা মানে মূলত অপারেটিং সিস্টেমের কাছে জানিয়ে দেওয়া, বস, দিস নিডস অ্যাকশন। যেমন ধরুন ক্রন ডিমন, তাকে মা-কালী বলে ডাকাই যায়, কালক্রম-অনুসরণ বা কাল-ই তার কাজ (কার্টসি রাঘব বন্দোপাধ্যায়), সময়ের টাইমার সার্কিটের দিকে তাকিয়ে তাকিয়ে তার আঁখি না-ফিরে। গ্নু-লিনাক্স অপারেটিং সিস্টেমে এরকম অনেক কাজ আছে যা সময় মেপে করতে হয়। চাইলে আপনি নিজেও কাজ দিতে পারেন কালী যখকে। আপনি অপারেটিং সিস্টেম নন বলে যে আপনার কাজটা রিফিউজ করবে, এরকম নীচ

মনই নয় তার। এবার ধরুন, আপনি বলে দিয়েছেন, ঠিক এই সময়ে এই কাজটা দরকার আপনার। সেই সময়টা আসা মাত্রই কালী যখ জানিয়ে দেবে অপারেটিং সিস্টেমকে, কাজের সময় হল।

এরকম অনেক যখ আছে আপনার সিস্টেমে, আরো অনেক কিছু আছে। আপনি তাদের কথা জানুন আর না-জানুন, আপনার মেশিনে কার্যরত জ্যান্ত ব্যবস্থাটার গভীর গোপন অন্দরে তারা ছেদহীন যতিহীন ত্রুটিহীন ভাবে কাজ করে চলেছে। পরে আমরা দেখব, এরা যে আছে, সেই উপস্থিতিটাকে ফুটিয়ে তোলার দেখার নানা উপায়। যেমন একটা হল, মেশিনের মধ্যে একটা কোনো মুহূর্তে চলমান প্রতিটি প্রক্রিয়ার একটা তালিকা নেওয়া। গ্নু-লিনাক্স সিস্টেমে তার জন্যে কমান্ড বা আদেশ দেওয়া যায়, ‘`ps aux`’। এরকম যে কোনো কমান্ডই কমান্ড প্রম্পটে দেওয়ার পরে এন্টার মারলে কম্পিউটার সেই আদেশটা পালন করে। এই কমান্ডটুকুর মধ্যে ‘ps’ অংশটা হল মূল আদেশ, আর ‘aux’ তার অপশান বা বিশেষ নিয়ন্তা। এগুলো আমরা পরে বুঝব, কমান্ড, কমান্ডের গঠন, তার অপশন ইত্যাদি। ‘ps’ কমান্ডটা স্ক্রিনে একটা তালিকা দেখায় চলমান প্রক্রিয়াগুলোর বা প্রসেসের। পরে আমরা দেখব এই এক একটা কাজকে ডাকা হয় এক একটা প্রসেস বলে। আমরা আগেই বলেছি প্রতিটি প্রোগ্রামই এক একটা ফাইল। একটা বিশেষ ধরনের ফাইল যারা নিজেরাই চলে। সক্রিয় বা এক্সিকিউটেবল বা বাইনারি ফাইল। এই ধরনের একটি বিশেষ ফাইলের একবার চলাকেই বলে একটা প্রোগ্রাম রান করা। একটি বিশেষ প্রোগ্রামের কোনো একবারের ক্রিয়াশীলতাকে বা রান-করা-কে আমরা ডাকি একটা প্রসেস বা প্রক্রিয়া বলে। একই সফটওয়ার প্রোগ্রামকে যদি দুবার আলাদা করে চালানো হয়, সেটাকে ধরা হবে দুটো প্রক্রিয়া। কোনো একটা প্রোগ্রাম যখন নিষ্ক্রিয় বা স্থির রয়েছে তখন সেটা ফাইল, যখন সে ক্রিয়াশীল, রান করছে, সেটা একটা প্রসেস বা প্রক্রিয়া। এই প্রসেস বা প্রক্রিয়াকে পরে আমাদের আরো সক্রিয় ভাবে বুঝবে — তার কথায় আমরা পরে আসছি।

প্রসেস বা প্রক্রিয়া বলতে এক কথায় ধরে নিন, একটা চলন্ত প্রোগ্রাম। ডিমন বা যখও একটা প্রসেস, আবার আপনি একটা গান বাজানোর অ্যাপ্লিকেশন চালাচ্ছেন, সেটাও প্রসেস, ওয়ার্ড প্রসেসর খুলে লিখছেন, বা জিসিসি দিয়ে কম্পাইল করছেন — এরা প্রত্যেকেই এক একটা প্রসেস। এই প্রসেসরা প্রত্যেকেই ব্যবহার করছে সিপিইউ বা প্রসেসরকে, মেমরিকে এবং কনসোল ইত্যাদি নানা ধরনের আইও মাধ্যমকে। অথচ রসদ তো অনন্ত না, একটা কম্পিউটারের মোট প্রসেসর সামর্থ কী, র‍্যাম কতটা, হার্ডডিস্ক কী সাইজের, প্রিন্টার কটা, কনসোল কী মানের — এরকম সবকিছুই তো প্রদত্ত, সেই মুহূর্তে অপরিবর্তনীয়। তাহলে এই সীমিত সামর্থকে ব্যবহার করেই কাজ করতে হবে প্রতিটি প্রক্রিয়ার। তাই, এদের মধ্যে নিয়তই একটা প্রতিযোগিতা চলছে — কোন প্রসেস কোন রিসোর্সকে কতটা সময়ের জন্যে ব্যবহার করতে পারবে। প্রসেসর, মেমরি, আইও — এই বিবিধ রসদকে একটা নিয়ন্ত্রিত এবং সংগঠিত রকমে বিভিন্ন চলমান সফটওয়ারের কাছে, মানে প্রসেসের কাছে, পৌঁছে দেওয়াই অপারেটিং সিস্টেমের কাজ। নিয়ন্ত্রিত এবং সংগঠিত রকমে। যাতে সব কটা অ্যাপ্লিকেশন তথা প্রক্রিয়াই তাদের নিজের নিজের কাজ করে চলতে পারে, তাদের গুঁতোগুঁতি না-করতে হয়, আবার সব দুধ শুধু ছাগলের প্রথম দুটো ছানাই খেয়ে গেল, তৃতীয়টা শুধু নেচেকুঁদে বেড়াল, এরকমটাও যাতে না-হয়।

ধরা যাক, তিনটে সফটওয়ারই তাদের কাজ একইসঙ্গে মনিটরে বা পর্দায় পৌঁছে দিতে চাইছে। কী দাঁড়াবে বিষয়টা? একটু সাইফি মানে সায়েন্স ফিকশন করা যাক। পর্দা জুড়ে, ধীরে, ঘুমন্ত বাচ্চার চোখ খোলার মত করে, ছোট ছোট কুচি কুচি যেগুলো ছড়িয়ে পড়ল প্রস্ফুটিত গোলাপ-পাঁপড়ির লাস্যময় শরীরে — না, সেগুলো কিন্তু পরাগ না, আপনার কবিতার যতিচিহ্ন — কোলন, সেমিকোলন, কমা। আর তিরতির করে নড়তে নড়তে এপাশ থেকে ওপাশ চলে যাচ্ছে, উদাসীন, ঋজুরেখ — সে বা তারা কোনো মধুলোভী মধুকরও নয়, নিতান্তই কম্পাইলেশনকালীন কম্পাইলারবার্তা। কবিতা, গোলাপ আর কম্পাইলেশন এখন জমাট দ্রবীভূত এক। মানে? মানে কেলেঙ্কারি। ক্যাওস। সাড়ে-বত্রিশ-ভাজা। তিনটে কাজকে তিনটে বাফারে বা অস্থায়ী মেমরিভূমিতে সংহত করে এটাকে একটা সংবদ্ধ বিস্রস্ত গোছালো জায়গায় নিয়ে আসাটা আশু প্রয়োজন, আর ঠিক সেই কাজটাই করে অপারেটিং সিস্টেম।

যখনি একটা কাজ শেষ হল, ধরুন কবিতাটা, তারা স্থান পেয়ে গেল আপনার কবিতাদের জন্য নির্দিষ্ট কোনো বিশেষ নামের ডিরেক্টরিতে, সেই এক বা একাধিক কবিতার জন্য ন্যস্ত একটি বিশেষ নামের ফাইলে। এবং এই পুরো সময়টা জুড়ে অন্য কাজগুলোকে কোনো ধরনের কোনো বিরক্তিই কিন্তু করা হচ্ছে না। ভাই, আমার সিস্টেমের রসদ দিয়ে আমি এখন কবিতার ফাইল লিখছি, তোমরা অন্য অন্য ছাগশিশুগন একটু দৈত্যের বাগানে পর্যটন করে এসো,

এরকমটা কিন্তু বলে দিচ্ছেনা অপারেটিং সিস্টেম। সবাই, প্রত্যেকেই ঘটে চলেছে, প্রতিমুহূর্তেই, একবারো না-থেমে। এই লিভ অ্যান্ড লেট লিভের গনতন্ত্রটাই ঘটছে অপারেটিং সিস্টেমের দৌলতে।

অনেকগুলো সমান্তরাল ভাবে চলমান প্রক্রিয়ার এই জটিল কেচ্ছাটা আরো কেলোপরায়ণ হয়ে ওঠে যখন সিস্টেমটা একটা একটা বহু-ব্যবহারকারী বা মাল্টিপল-ইউজার সিস্টেম। আর আগেই তো বলেছি, যে কোনো গ্নু-লিনাক্স বা ইউনিক্স সিস্টেমই আভ্যন্তরীণ ভাবে সেরকম করেই তৈরি। ধরুন এগারো জন ব্যবহারকারী এগারোটা টার্মিনাল থেকে একত্রে লগ-ইন করেছেন সিস্টেমে, একই সঙ্গে লিখছেন এগারো রকমের টেক্সট, কবিতা থেকে হালখাতা। এবং এই টেক্সট লিখে-চলা-কালীন তারা চালাচ্ছেন তিনটে করে প্রোগ্রাম, মানে এগারো গুণ তিন তেত্রিশটা সফটওয়ার চালাচ্ছেন মোট। তেত্রিশখানা জীবন্ত ধাবমান প্রসেস। এর সঙ্গে আছে গোদের উপর বিষফোঁড়ার মত অগুন্তি সিস্টেম প্রসেস, সিস্টেমের কাজ করে চলার জন্যেই অত্যাবশ্যক কিছু প্রক্রিয়া, ওই ডিমন বা যখদের মত। আর এই এগারো জনের ইউজার টিম, এরা প্রত্যেকেই তাদের রচনাকে কম্পিউটার মেমরিতে ধৃত নিরালম্ব বায়ুভূত ইলেকট্রনিক থেকে বাস্তব কাগজের আর কালির প্রিন্টের আকারে নিয়ে আসছেন একমেবাদ্বীতিয়ম নেটওয়ার্কাবদ্ধ প্রিন্টার থেকে। এরকম একটা অবস্থায় এই টিমের একজন হিশেবে আপনি যেই নিজের কবিতার প্রিন্টআউট চাইলেন, সেই শুদ্ধ ভূর্জপত্রে আপনার কোনো গোলাপোল্লাসের পরের লাইনেই যে, ‘গরুর খোল উনিশ মন’, বা জিসিসির কোনো বার্তা ছাপা হচ্ছে না, তার একমাত্র কারণই ওই অপারেটিং সিস্টেম। অপারেটিং সিস্টেমই পৌঁছে দিচ্ছে গরুকে তার গোয়ালে, চারি ভর্তি খোলকে গরুর চোয়ালে, এবং গোলাপকে ফুলদানিতে।

বহু ব্যবহারকারী। বহু ফাইল তাদের সম্পূর্ণ নিজস্ব। বহু ফাইল সর্বসাধারনের। বহু ফাইল আবার একমাত্র রুট বা ওই সিস্টেমের একচ্ছত্র শাসকের। ইউনিক্স বা যে কোনো গ্নু-লিনাক্স সিস্টেমে এরকমই নিয়ম, উইন্ডোজ এক্সপি-র বেলাতেও অনেকাংশে যা সত্যি। এই অজস্র ধরনের মালিকানার ফাইলের উপর অজস্র ধরনের অধিকার, সেগুলোর উপর একই সঙ্গে কাজ করছে বহু ব্যবহারকারীর চালু করা বহু প্রক্রিয়া, এরা প্রত্যেকেই দৌড়চ্ছে মানে রান করছে একই সঙ্গে। এর সঙ্গে রয়েছে অপারেটিং সিস্টেমের নিজের শুরু করা নিজের জন্যেই প্রয়োজনীয় অনেকগুলো প্রক্রিয়া। অপারেটিং সিস্টেমকে তাই প্রতিমুহূর্তে হিশেব করতে হচ্ছে কোনো একটা বিশেষ সময়বিন্দুতে কোন রসদকে ব্যবহার করছে কোন ব্যবহারকারীর কোন প্রক্রিয়া। সে রয়েছে যাবতীয় রসদের ভাঁড়ারের দায়িত্বে।

৫।। সমান্তরাল বহুক্রিয়া বা মাল্টিপ্লেক্সিং

পরে আমরা দেখব নির্দিষ্ট পরিমাণ রসদকে এই বিবিধ প্রক্রিয়ার মধ্যে ভেঙে দেওয়ার জন্যে অপারেটিং সিস্টেমের একটা নিজস্ব পদ্ধতি আছে। চলমান সমস্ত প্রক্রিয়া বা প্রসেসের মধ্যে সীমিত রসদ শেয়ারিং বা ভাগবাটোয়ারার যে পদ্ধতির টেকনিকাল নাম মাল্টিপ্লেক্সিং। দুটো আলাদা আলাদা তলে এই বণ্টনটা ঘটে। একটা তল হল ভূমি বা স্পেস। অন্য তলটা কাল বা টাইম।

মাল্টিপ্লেক্সিং বা বহুক্রিয়তা (জঘণ্য শোনাচ্ছে, কিন্তু এর চেয়ে ভালো কিছু মাথায় আসছে না প্রতিশব্দ হিশেবে, এর পর থেকে মাল্টিপ্লেক্সিং-ই লিখব) গোটা ব্যাপারটা অত্যন্ত জটিল। আমি লিখব কী, আমার নিজেরই আন্দাজ অত্যন্ত ভাসাভাসা, যতটুকু না-জানলে গ্নু-লিনাক্সের মাল্টি-টাস্কিং বা একই সঙ্গে একাধিক কাজ করতে পারার বাড়তি সক্ষমতাটাকে ঠিক ভাবে বুঝে ওঠা যায়না। আসলে আমার নিজের গ্নু-লিনাক্স অভিজ্ঞতার প্রথম বিস্ময়গুলোর একটা ছিল এটাই। তার আগেকার অনেকগুলো বছরের ডস-উইন্ডোজ করতে গিয়ে একভাবে অভ্যস্ত ছিলাম। সিনেমায় বা তথ্য চিত্রে যখন দেখতাম একই সঙ্গে কতগুলো করে উইন্ডোজ খোলা, কিম্বা, অনেকগুলো করে কাজ একই সঙ্গে হচ্ছে, ভাবতাম ওগুলো সিনেমার ম্যাজিক, সেই লাস্ট অ্যাকশন হিরো সিনেমার বাচ্চাটা শোয়ার্জনেগারকে যেমন বলেছিল, ‘দেখে বুঝতে পারছ না, এটা বাস্তব জীবন নয় সিনেমা — দেখছ না চারদিকের প্রত্যেকটা মেয়েই কেমন এলিগ্যান্ট’। বা ভাবতাম, প্রথম বিশ্বের প্রথম শ্রেণীর হার্ডওয়ারের ক্যালি। গ্নু-লিনাক্স করার প্রথম দিকে সিস্টেমের মধ্যে দেওয়া হাউ-টু পড়ে যখন দেখতাম, অনেক ভালো মাল্টিটাস্কিং-এর কথা বারবার উল্লিখিত, তখনো বুঝিনি, ব্যাপারটা আসলে কতদূর সত্যি। এখন তো অভ্যস্ত হয়েই গেছি, একই সঙ্গে অনেকগুলো কাজ চালিয়ে, কিন্তু সেটা উইন্ডোজ অপারেটিং সিস্টেমের নিরিখে এতটাই অবিশ্বাস্য যে আমি আলাদা করে বলতে যাচ্ছি না, নিজেই করে দেখুন। আমার মেশিনের হার্ডওয়ার খুব একটা অগ্রবাহিনী নয় আদৌ, অ্যাথলন ১৭০০ প্রসেসর, এএন২৬৬ভিএম

মাদারবোর্ডে, ২৫৬ এমবি র‍্যাম, আশি জিবি হার্ডডিস্ক, এবং দুটো সিডি ড্রাইভ, একটা রাইটার একটা রিডার, একটা ডটম্যাট্রিক্স প্রিন্টার। তাতেই এতকিছু করা যায়। একটু ভালো হার্ডওয়ার লাগালে মনে হয় একসঙ্গে চারপাঁচটা বই লেখা যেত।

যখন একটা বিশেস রসদকে সময় বা টাইম-এর নিরিখে মাল্টিপ্লেক্স করা হয়, বিভিন্ন আলাদা আলাদা প্রক্রিয়া তখন খাপ পেতে থাকে, কখন আগের প্রক্রিয়াটা রসদটাকে ছেড়ে দেবে, এবং সে রিসোর্সটার উপর নিজের অধিকার কায়েম করবে। প্রতি সেকেন্ড সময়কে অনেকগুলো ছোট ছোট টুকরোয় ভেঙে ফেলা হয়। আগেই বলেছিলাম, আপনি আপনার মেশিনের অপারেটিং সিস্টেমের কাছে ব্রহ্মা বললেই হয়, আপনার চোখের পাতা একবার পড়তে পড়তে তার হাজার যুগ চলে যায়। সময়ের এই অজস্র ছোট ছোট টুকরোর এক একটা টুকরো জুড়ে একটা বিশেষ রসদ থাকে একটা বিশেষ প্রক্রিয়ার দখলে। তার পরের সময়ের টুকরোয় ওই রসদটা চলে যায় পরের প্রক্রিয়ার হাতে। প্রথম প্রক্রিয়ার পরে পায় দ্বিতীয়, তারপরে তৃতীয়, এরকম ঘুরতে থাকে। শেষ প্রক্রিয়াটার পরে আবার ঘুরে আসে প্রথম প্রক্রিয়া।

এটাকে মাল্টি-টাস্কিং বলেও ডাকা হয় — তবে সেটা এই অর্থে যে একই সঙ্গে কম্পিউটার একাধিক টাস্ক-এ বা কাজে রত থাকছে। যেমন ধরা যাক, একটি এক-প্রসেসর কম্পিউটার সিস্টেমে (একটা কম্পিউটারে প্রসেসরের সংখ্যা একের বেশি হতেই পারে, যেমন সিমেট্রিক-মাল্টি-প্রসেসিং বা এসএমপি, যদিও সচরাচর, বিশেষত পিসি বা পার্সোনাল কম্পিউটারের বেলায় আমরা একটা প্রসেসরের মেশিনেই অভ্যস্ত) প্রথম সময়-টুকরোটা পাচ্ছে প্রথম প্রক্রিয়া — সেই সময়টুকু জুড়ে প্রথম প্রক্রিয়া ব্যবহার করে নিচ্ছে প্রসেসরটাকে। পরের সময়-টুকরোয় প্রসেসর চলে যাচ্ছে দ্বিতীয় প্রক্রিয়ার দখলে। এই ভাবে ঘুরছে। কোন প্রক্রিয়া কত সময় ধরে প্রসেসরকে পাবে, যার পর তাকে ছেড়ে দিতে হবে — এটা ঠিক করে দেওয়া-ই অপারেটিং সিস্টেমের কাজ। প্রিন্টারও কাজ করে অনেকটা এই ভাবেই। যখন অনেকগুলো প্রিন্ট জব বা ছাপার কাজ লাইনে আছে — সকলেই অপেক্ষা করছে প্রিন্ট হবে প্রিন্ট হবে বলে, প্রিন্টারের স্পুলদেহে আসিতেছে চলে, প্রিন্ট হয়ে আসিয়াছে যারা, প্রিন্টিত হতে হয় যাহাদের, এবং এতগুলো শবরী-প্রিন্টজবের পাথর ছুঁয়ে দেওয়ার জন্যে রাম মাত্র একজন, মানে প্রিন্টারের সংখ্যা এক — তখন অপারেটিং সিস্টেমই সিদ্ধান্ত নেয় কোন প্রিন্ট কার পরে এবং কখন শুরু হবে, কার পরে কে আসবে।

অন্য ধরনের মাল্টিপ্লেক্সিং-টা হল ভূমি বা স্পেস ভিত্তিক মাল্টিপ্লেক্সিং। একটা লাইনে দাঁড়ানো খদ্দেররা একের পর এক আসছে — একজনের হয়ে গেলে পরের জন, এবং প্রতিটি খদ্দেরেরই চাহিদা অশেষ, তারা একটা মাল পাওয়া মাত্রই ঘুরে গিয়ে আবার লাইনে দাঁড়িয়ে পড়ছে পরের মালের জন্যে — কালভিত্তিক মাল্টিপ্লেক্সিং-এ এই ভাবে কাজ হচ্ছিল। এবার পদ্ধতিটা আর সময়ের নিরিখে হচ্ছে না, হচ্ছে ভূমির নিরিখে। মানে রসদটাকে একটা জমি বলে ভাবুন, মেশিনের প্রাপ্তব্য মোট জমি। এবং যখনই কোনো প্রক্রিয়া তার ঘর বাঁধতে চাইছে তখনই ওএস মানে অপারেটিং সিস্টেম (OS — **O**perating-**S**ystem) তাকে সেই মোট জমির থেকে একটুকরো জমি বরাদ্দ করছে, এখন থেকে এই জমিটুকু তোমার। এই ঘর-বাঁধাটা কখনো টেম্পোরারি কখনো পার্মানেন্ট। কোনো প্রক্রিয়া হয়তো কয়েক লহমার একটা টেম্প বা অস্থায়ী ফাইল লিখেই খুশি, তার কাজ শেষ হলেই উবে যাচ্ছে কাগজের ঘর, জমি ফেরত চলে যাচ্ছে ওএস-এর রাজকোষে। আর কোনো প্রক্রিয়া হয়তো বের হয়ে আসার আগে তার সমস্ত ফলাফল ছেপে দিচ্ছে কাগজের পাতার প্রিন্টে বা সিডির অপটিকাল চাকার পাতায় লেজারের আখরে।

এই ভূমি ভিত্তিক মাল্টিপ্লেক্সিং ব্যবস্থায় প্রতিটি খদ্দেরই একটা রিসোর্সের এক একটা টুকরোকে পেয়ে যাচ্ছে। সময়কে টুকরো না করে এখানে টুকরো করা হচ্ছে রসদটাকে। যেমন মূল র‍্যান্ডম-অ্যাকসেস-মেমরি বা র‍্যামকে ভাঙা হয় এই ভাবেই। একটা সময় বা কালভিত্তিক মাল্টিপ্লেক্সিং পদ্ধতির কথা ভাবুন। পদ্ধতিটা চলছে, মানে, একই সঙ্গে জ্যান্ত একাধিক প্রক্রিয়া চলছে মেশিনে। এবং একটু বাদে বাদে এক এক টুকরো সময় জুড়ে সিপিইউকে ব্যবহারের অধিকার এক একটা প্রোগ্রামের হাতে ছেড়ে দিচ্ছে ওএস। এবার ভাবুন, এটা যদি চালিয়ে যেতে হয়, তাহলে এই কালভিত্তিক মাল্টিপ্লেক্সিং-এর স্বার্থেই একটা ভূমিভিত্তিক মাল্টিপ্লেক্সিং পদ্ধতি-ও চালাতে হবে ওএস-কে। একাধিক ক্রিয়াশীল প্রোগ্রাম বা প্রসেসের জন্যে মোট মেমরিটাকে টুকরোয় টুকরোয় ভেঙে ফেলতে হবে। যাতে একই সঙ্গে একাধিক প্রোগ্রাম র‍্যামে থাকতে পারে। কারণ একটু বাদে বাদেই তো প্রত্যেকের-ই পালা আসছে প্রসেসর বা সিপিইউ ব্যবহার করার, কালভিত্তিক মাল্টিপ্লেক্সিং-এর নিয়ম মেনে। আর কোনো একটা প্রোগ্রাম যদি সিপিইউ-কে

ব্যবহার করতে চায়, ওই প্রোগ্রাম চলার জন্যে প্রয়োজনীয় গোটা তথ্যটাকেই তো মেমরিতে থাকতে হবে — যাতে চাওয়া মাত্র সিপিইউ তাকে মেমরিতে খুঁজে পেতে পারে। তাই সবকটা প্রোগ্রামকেই একই সঙ্গে মেমরির মধ্যেই বসবাস করতে হবে, অল্প অল্প জমি নিয়ে, তেঁতুল পাতায় ন-জনের মত। একবার ভেবে দেখুন, বাড়তি ভৌতিক স্মৃতি বা ভারচুয়াল মেমরির প্রয়োজন পড়ে এখান থেকেই। সীমিত র‍্যামে যখন গোটাটা আর রাখা যাচ্ছেনা, ওএস তখন আর একটা বাড়তি জমিকে অস্থায়ী রকমে র‍্যামের প্যালা হিশেবে নিয়ে আসছে। এবার র‍্যামে তাদের কানটা রেখে দেওয়া হচ্ছে, যাতে প্রয়োজন পড়লেই সিপিইউ র‍্যামে থাকা তাদের কান ধরে টেনে-ই ভারচুয়াল মেমরি বা ভৌতিক স্মৃতি থেকে তাদের মাথাটা পেয়ে যেতে পারে।

অন্য আর এক ভাবেও এই গোটাটা করা যেতে পারত। তা হল গোটা মেমরিটাই একটা প্রোগ্রামকে দিয়ে দেওয়া, তারপর, তার পালা শেষ হলে আবার গোটা মেমরিটাকেই পরের প্রসেস বা প্রক্রিয়ার জন্যে ছেড়ে দেওয়া। বেশিরভাগ ক্ষেত্রেই, যদি পর্যাপ্ত র‍্যাম থাকে, এই দ্বিতীয় পদ্ধতিটা অর্থহীন। কারণ, স্বাভাবিক একটা ওএস-এ, স্বাভাবিক অবস্থায়, বেশির ভাগ প্রোগ্রামেরই সাইজ যা তাতে গোটা র‍্যামের কোনো প্রয়োজনই পড়ে না। গোটা র‍্যামের একটা ছোট্ট ভগ্নাংশতেই তারা ধরে যায়। আর একটা রসদ যাকেও এই একই ভাবে ভূমি বা স্পেসগত মাল্টিপ্লেক্সিং করা হয় — সেটা হল হার্ডডিস্ক। একই সঙ্গে একই হার্ডডিস্কের শরীরে খচিত থাকে অজস্র প্রোগ্রাম ফাইল এবং তাদের কাজ করার জন্যে প্রয়োজনীয় আরো প্রচুর ফাইল। একই সঙ্গে ক্রিয়াশীল একাধিক প্রসেস তাদের কাজের ফাইলগুলো লিখতে এবং পড়তে থাকে একই হার্ডডিস্কে। এবং গ্নু-লিনাক্স বা ইউনিক্স এর বেলায় এর এক একটা প্রসেস হয়ত চালাচ্ছে এক একজন ইউজার। হার্ডডিস্কের কোন অংশটা কখন কোন প্রসেসের জন্যে বর্তাবে সেটা ঠিক করে দেয় এই ওএস বা অপারেটিং সিস্টেম। কী ভাবে করে সেটা আমরা জানব ছয় এবং সাত নম্বর দিনে ফাইলসিস্টেম জানার সময়।

৬।। আইও ডিভাইস, ডিভাইস কন্ট্রোলার এবং ইন্টারাপ্ট

আগের সেকশনটা দেখুন, আমরা রসদের উদাহরণ দিতে গিয়ে বারবার নানা ধরনের স্মৃতির কথাই বলেছি মূলত। মেশিনের মূল স্মৃতি র‍্যাম, বা হার্ডডিস্ক জাতীয় নানা ধরনের স্মৃতির ভাঁড়ার বা স্টোরেজ ডিভাইস। কিন্তু রসদ মানে কিন্তু আদৌ শুধু স্মৃতি নয়। এর মধ্যে পড়ে নানা ধরনের আইও বা ইনপুট-আউটপুট ডিভাইস। যার বেশ কিছু উদাহরণ ইতিমধ্যেই দিয়েছি আমরা, আজ এবং শূন্য নম্বর দিনে। অপারেটিং সিস্টেমের এই রসদ-বন্টন রাজত্বের খাস প্রজা অবশ্যই মেমরি। তার প্রসঙ্গে সামান্য কিছু কথা আমরা আগেই বলেছি, এক নম্বর দিনের গোড়ায়। আরো পরে আসব।

আজকের আলোচনার ২ নম্বর সেকশনের ছবিটা একবার দেখে নিন। যেখানে আমরা মেমরি এবং প্রসেসরের সঙ্গে পেরিফেরাল উপাদানগুলোর বাস-সংযোগের ছকটা দেখিয়েছিলাম। আমরা এখন সামান্য কিছু কথা বলে নেব এই আইও উপাদান এবং বাস নিয়ে। মেমরির পরেই অপারেটিং সিস্টেমের সবচেয়ে বড় ব্যস্ততাটা থাকে এই ইনপুট-আউটপুট নিয়ে। বাস-সংযোগের ছবিটায় দেখুন, প্রতিটি আইও উপাদানের মূলত দুটি অংশ — একটা হল মূল উপাদান বা তার ভৌত শরীর, এবং অন্য অংশটা হল তার নিয়ন্ত্রণ। কনসোল, কিবোর্ড, ফ্লপি-ডিস্ক এবং হার্ডডিস্ক — এই চারটেকে দেখিয়েছি আমরা ছবিতে, এরকম আরো হতে পারে। ধরুন, ফ্লপি ডিভাইস বলতে আমরা বুঝি ফ্লপি ঢোকানোর এবং চালানোর ড্রাইভটা, এই ভৌত যন্ত্রাংশটা। আর তার নিয়ন্ত্রণটা ঘটে ফ্লপি-ডিস্ক কন্ট্রোলার বলে একটা জিনিষ দিয়ে। এই কন্ট্রোলার হল একটা চিপ সমেত সার্কিট। বা অনেক ক্ষেত্রে কয়েকটি চিপের মিলিত একটা কার্ড বা প্লাগ-ইন-বোর্ড, যাদের ছবি আমরা শূন্য নম্বর দিনে দেখিয়েছি। এই চিপ বা চিপপুঞ্জই ভৌত ড্রাইভটায় তথ্য এবং বিদ্যুৎপ্রবাহের যাতায়াতকে নিয়ন্ত্রণ করে। অপারেটিং সিস্টেমের আদেশ মোতাবেক ড্রাইভকে সক্রিয় করে তোলে এই কন্ট্রোলার এবং আদেশ-নির্দিষ্ট তথ্য ফ্লপি-ডিস্ক এবং সিপিইউ-র মধ্যে দেওয়া নেওয়া করে। যখন ফ্লপি-ডিস্কে তথ্য লেখা হয় তখন তথ্যটা যায় সিপিইউ থেকে ফ্লপি-ড্রাইভ, আর যখন ফ্লপি-ডিস্ক থেকে তথ্য পড়া হয়, তখন তথ্য যায় ফ্লপি-ড্রাইভ থেকে সিপিইউ। নিয়ন্ত্রণ করার জন্যে এই কন্ট্রোলার সার্কিট এবং মূল ভৌত যন্ত্রাংশটা — এই দুটো জিনিষ মিলে তৈরি হয় মেশিনের ওই ড্রাইভটা। শুধু ফ্লপি না প্রতিটি ড্রাইভের বেলাতেই তাই।

ড্রাইভের আদত নিয়ন্ত্রণটা অত্যন্ত জটিল গোলমেলে এবং ঘোরপ্যাঁচসঙ্কুল ইলেকট্রনিক একটা ব্যাপার। তাকে একটা সহজতর আকারে অপারেটিং সিস্টেমের কাছে হাজির করে এই কন্ট্রোলার। যেমন, আপনি হয়ত হার্ডডিস্ক থেকে একটা বিশেষ নামের ফাইল খুলতে চাইলেন, পড়বেন বলে। অপারেটিং সিস্টেম তার আইনোড তালিকা পড়ে সেই ফাইলের নামের সঙ্গে মিলিয়ে নিল, তারপর হার্ডডিস্ক ডিভাইস কন্ট্রোলারকে জানাল, এত এত নম্বর সেক্টর থেকে তথ্য পড়ে আমায় পাঠাও। হার্ডডিস্ক ডিভাইস কন্ট্রোলার তথ্যটা পড়ে সিপিইউ-র কাছে পাঠিয়ে দিল। অপারেটিং সিস্টেম তখন সিপিইউ-কে দিয়ে সেই তথ্যটা ফের পাঠিয়ে দিল কনসোলের ডিভাইস কন্ট্রোলার মানে ভিডিও কন্ট্রোলারের কাছে। ভিডিও কন্ট্রোলারের নিয়ন্ত্রণে ভিডিও ডিভাইস তখন সেই তথ্যটা তার শরীরে মানে কনসোলে ফুটিয়ে তুলল, এবং আপনি ফাইলটা পড়লেন।

এই গোটা কাজটার ভিতরে নিহিত জটিলতাটাকে খেয়াল করুন। ডিভাইস কন্ট্রোলারগুলোর কল্যাণে গোটা কাজটা কত সহজ হয়ে যাচ্ছে। অপারেটিং সিস্টেমের পাঠানো ওই সেক্টর নম্বরটা থেকে কন্ট্রোলার হিশেব কষে ফেলল এত নম্বর সেক্টর মানে কত নম্বর সিলিন্ডার, কোন জায়গা, কোন হেডকে কতটা ঘুরিয়ে সেখানে পাঠাতে হবে — ইত্যাদি (এই সিলিন্ডার ইত্যাদি আর একটু বিশদ ভাবে আমরা জানব যখন ফাইল সিস্টেমের আলোচনায় ঢুকব, মানে আট নম্বর দিনে গিয়ে)।

এখানে জটিলতা আরো একটু বাড়তে পারে এই ভাবে যে হার্ড ডিস্কে হয়ত কিছু ব্যাড সেক্টর দেখা দিয়েছে, যেখানে আর তথ্য পড়া বা লেখা যাবেনা, সেই সেই সেক্টরের তথ্য তুলে নিয়ে হার্ডডিস্ক কন্ট্রোলার অন্য নির্দিষ্ট কিছু সেক্টরে রেখেছে। এই হিশেবটাও মাথায় রাখতে হবে কন্ট্রোলারকে। এবার তাকে হিশেব করতে হবে ওই হেডকে ওই জায়গায় পাঠানো মানে কতটা নড়ানো, কতটা বিদ্যুৎপ্রবাহ লাগবে তাতে — ইত্যাদি। ফাইল পড়ার পুরো প্রক্রিয়াটা জুড়েই নিরবচ্ছিন্ন ভাবে এই কাজটা করে চলতে হবে কন্ট্রোলারকে। এবং একই সঙ্গে, যত যত বিট সে পড়ছে — সেই তথ্যটাকে ওই কম্পিউটারের ওয়ার্ডের মাপে দুই বা চার বা আট বাইট করে পাঠাতে থাকবে অপারেটিং সিস্টেমের কাছে। এই গোটা কাজটা করে চলে কন্ট্রোলারের সার্কিট বোর্ডের যাবতীয় উপাদানগুলো সকলে মিলে। ডিভাইস কন্ট্রোলারগুলো না-থাকলে তাই ওএস-এর কাজটা বহুগুণ জটিলতর হয়ে উঠত।

যা বললাম, এই কন্ট্রোলারকে বাদ দিয়ে ওই পেরিফেরাল ডিভাইসের আর একটা অংশ হল ডিভাইসের নিজেরই ভৌত শরীরটা। কন্ট্রোলারের তুলনায় ডিভাইসের নিজের জটিলতাটা বরং অনেক কম। তার একটা কারণ যে তাদের খুব একটা কিছু নানা ধরনের কাজ করতে হয়না, আর দ্বিতীয় কারণ, এই ধরনের ডিভাইসগুলোর সমস্ত নির্মাতাকেই কিছু স্বীকৃত মান বা স্ট্যান্ডার্ড মেনে নিতে হয়। যে স্ট্যান্ডার্ড অনুযায়ী এদের বানানো হয় — নইলে আলাদা আলাদা কোম্পানির ডিভাইস আলাদা আলাদা মেশিনে কাজ করানো যাবেনা। যেকোনো কোম্পানির আইডিই (IDE — **I**ntegrated-**D**rive-**E**lectronics) ডিভাইস যাতে অন্য যে কোনো কোম্পানির কন্ট্রোলার-এর সঙ্গে লাগানো যায়। পেন্টিয়াম অ্যাথলন এই জাতীয় সমস্ত মেশিনেই লাগানো হয় এই আইডিই ডিস্ক — আমরা যে ধরনের হার্ডডিস্ক নিয়েই সচরাচর কাজ করি। মূল ডিভাইসের ভিতরকার জটিলতাটাকে আমাদের প্রায় কখনোই আলাদা করে মুখোমুখি হতে হয়না এই কন্ট্রোলারের দৌলতে।

এটা হল আইও ডিভাইসের হার্ডওয়ার নিয়ন্ত্রণ, এবার একটু আলগা ছুঁয়ে নেওয়া যাক এর সফটওয়ার নিয়ন্ত্রণের প্রসঙ্গ। হার্ডওয়ারের স্তরে, ইলেকট্রনিক ইলেকট্রিকাল এবং ফিজিকাল, বৈদ্যুতিন বৈদ্যুতিক এবং ভৌত স্তরে, প্রতিটি ডিভাইস এবং তার কন্ট্রোলারের গঠন অন্য প্রতিটি ডিভাইসের থেকে আলাদা। তাই স্বাভাবিকভাবেই এদের কাজ করার জন্যে লাগবে আলাদা প্রোগ্রাম বা সফটওয়ার। একেই বলে ডিভাইস ড্রাইভার। মাউসের জন্যে মাউস ড্রাইভার, কনসোলের জন্যে ভিডিও ড্রাইভার ইত্যাদি। এই সফটওয়ার এবার সরাসরি কথা বলে ওই কন্ট্রোলারের সঙ্গে। কন্ট্রোলার কার্ড যারা বানায় তারাই বানায় এই ড্রাইভার, বা অপারেটিং সিস্টেমের নিজের ইনস্টলার-এর সঙ্গেই দেওয়া থাকে এগুলো। আলাদা আলাদা অপারেটিং সিস্টেমের জন্যে আলাদা আলাদা ড্রাইভার।

আইও ডিভাইসগুলোকে তাদের ভৌত উপাদান, এবং সেই ভৌত উপাদানকে কাজে লাগানোর জন্যে কন্ট্রোলার তথা ডিভাইস ড্রাইভারের সফটওয়ারের কার্যধারার ভিত্তিতে মোটামুটিভাবে দুটো বড় ভাগে ভাগ করা যায়।

ব্লক ডিভাইস — তথ্য-আয়তন-ভিত্তিক উপাদান। এরা তথ্যকে নাড়াচাড়া করে কিছু নির্দিষ্ট আয়তনের এককে, যে এককটার নাম ব্লক। এই প্রতিটি ব্লকের নিজস্ব একটা ঠিকানা আছে, সূচক আছে — মানে অন্য ব্লক তথা তথ্য থেকে স্বাতন্ত্র আছে। সচরাচর এই ব্লকের মাপ হয় ৫১২ থেকে ৩২৭৬৮ বাইট। এর সবচেয়ে গুরুত্বপূর্ণ বৈশিষ্ট্য এই যে, কোনো একটা নির্দিষ্ট ব্লক থেকে স্বতন্ত্র ভাবে তথ্য পড়া যায়, বা সেখানে লেখা যায়, অন্য ব্লকগুলোর লেজে কোনো পাড়া না-দিয়েই। অপারেটিং সিস্টেম চাইলে এক তিন পাঁচ এবং চৌষট্টি নম্বর ব্লকে তথ্য লিখতে পারে দুই চার এবং ছয় থেকে তেষট্টিকে নিষ্কলুষ শূন্য রেখে দিয়ে। ব্লক ডিভাইস নিয়ে এবং ব্লক এককে তার তথ্য নাড়াচাড়ার বিষয়টা নিয়ে আমরা অনেকটা আলোচনা করব আট নম্বর দিনে গিয়ে, ফাইলসিস্টেম নিয়ে আলোচনায়।

ক্যারেকটার ডিভাইস — তথ্য-চিহ্ন-ভিত্তিক উপাদান। ক্যারেকটার ডিভাইসের কাছে তথ্য মানে সারিবদ্ধ কিছু চিহ্ন বা ক্যারেকটারের প্রবাহ, এদের মোট আয়তন বা ব্লক সাইজ কী তা নিয়ে ক্যারেকটার ডিভাইসের বিন্দুমাত্র এসে যায়না। এখানে তথ্যের কোনো স্বতন্ত্র ঠিকানা বা অস্তিত্ব নেই। তাই এখানে আলাদা করে কোনো তথ্য একককে খোঁজা বা চিহ্নিত করা যায়না। প্রিন্টার, মোডেম, ল্যান কার্ড, মাউস জাতীয় পয়েন্টিং ডিভাইস — এরা সবাই হল ক্যারেকটার ডিভাইস। ডিস্ক জাতীয় নয় যে আইও ডিভাইসগুলো তারা সকলেই ক্যারেকটার ডিভাইস।

একটু ভালো করে ভাবলেই বোঝা যায় এই দুটো ধরনের আইও ডিভাইসের মধ্যে বর্গীকরণটা কোনো জল-অচল বর্গ বা ওয়াটারটাইট কম্পার্টমেন্ট নয়। ধরুন হার্ডডিস্ক যে ব্লক পদ্ধতিতে কাজ করে এটা খুব সহজেই বোঝা যায়। একটা হার্ডডিস্কের ভিতরে একটা হেড একটা সিলিন্ডারের গায়ে যে সেক্টরেই থাকুক না কেন সেই হেডটাকে নড়িয়ে বা অন্য কোনো হেডকে কাজে লাগিয়ে সহজেই অন্য একটা ব্লকে তথ্য লেখা যায় বা সেখান থেকে পড়া যায়। আলাদা আলাদা ঠিকানার ব্লককে চিনে নেওয়া যায় সহজেই।

এই সহজতাটা ঘেঁটে যায় যদি আপনি ক্যাসেট ড্রাইভকে ভাবেন। যেখানে একটা ক্যাসেটের ফিতেয় তথ্য তুলে রাখা হচ্ছে, বা সেখান থেকে পড়া হচ্ছে। সতত এই ক্যাসেট ড্রাইভ একটা ক্যারেকটার ডিভাইস হিশেবেই বিবেচ্য — প্রবহমান চিহ্নরাশির বীচিবিভঙ্গ খচিত করে রেখে চলেছে তার চৌম্বক শরীরে। এবার ভাবুন তো, একটা গোটা টেপময় তথ্যের ভাঁড়ারকে তো একটা একরৈখিক ব্লকের মিছিল বলে ভাবা যেতেই পারে — অপারেটিং সিস্টেম যখনই একটা বিশেষ অমুকতম ব্লকে তথ্য পড়তে বা লিখতে চাইবে, আরামসে সেটা সে করতে পারে, ক্যাসেটের ফিতেটাকে এগিয়ে পিছিয়ে, রিওয়াইন্ড আর ফাস্ট-ফরওয়ার্ড করে, যতক্ষণ না কোনো অনির্দেশ্য তমুক নয়, নিশ্চিতভাবেই ওই বিশেষ অমুকতম ব্লকটিকে সে খুঁজে পাচ্ছে। তার মানে ঠিক হার্ডডিস্কের মতই স্বতন্ত্র ঠিকানার ব্যক্তিস্বাতন্ত্র নিয়ে মুক্ত ব্লকতন্ত্র মানে ব্লকের গনতন্ত্র গড়ে তোলা যাচ্ছে ক্যাসেটের ফিতেতেই, শুধু একটু ইয়ে, মানে সময়টা একটু বড্ডই বেশি লাগছে। তাহলে ক্যারেকটারের ক্যারেকটার রইল কই — সে তো ব্লক হয়ে গেল?

কিন্তু এ প্রায় নব্যন্যায় হয়ে যাচ্ছে — বাঙালীর বিখ্যাত সেই তীব্র মেধাদীপ্ত সিরিয়াস ভাটের ট্র্যাডিশনের মত, বাঙালীর এই ক্যালি বোধহয় সম্রাট অশোকও জানতেন, ব্রাহ্মী শিলালিপিতে আছে গৌড়ীয় রীতির মানে কথাকে ফেনিয়ে বলার প্রথার উল্লেখ, বাজে বকাটা বোধহয় আমাদের বংশগত জাতিগত ইতিহাসগত — একটা যুক্তিকে তার লিমিটে পুশ করা — ক্যাসেট ড্রাইভ তো আর সত্যিসত্যিই ওভাবে ব্যবহার করা হয়না।

এটা ঠিক যে এই বর্গীকরণটা নিখুঁত নয়। কোনো কোনো ডিভাইস যেমন, ব্লক বা ক্যারেকটার কোনো খাপেই ঢোকানো যায়না। ক্লক বা ঘড়ি যেমন। এদের ব্লক-ঠিকানা দিয়ে ধরা যায়না, আবার এরা কোনো চিহ্নপ্রবাহ-ও পয়দা করেনা, ঘড়ি কি তবে কেউ নয়? সে তো যা করে সেটা হল কিছুক্ষণ অন্তর অন্তর একটা করে ইন্টেরাপ্ট মানে অপারেটিং সিস্টেমের উদ্দেশ্যে একটা করে সিগনাল তৈরি করে যাওয়া। তাহলে? এই ব্লক বা ক্যারেকটার ডিভাইস হিশেবে বর্গীকরণটা এইজন্যেই আছে যে এই ছকটা দিয়ে অপারেটিং সিস্টেমের আইও ডিভাইসদের নাড়াচাড়া করার প্রকৌশলগুলো বেশ ভালো ব্যাখ্যা করা যায়। এই কথাগুলো আনতে হল কারণ, পরে যখন আমরা গ্নু-লিনাক্স অপারেটিং সিস্টেমের ডিভাইস ফাইল এবং ফাইলসিস্টেম প্রসঙ্গে আলোচনা করব তখন এই পার্থক্যটা আমাদের দরকার পড়বে। ফাইলসিস্টেম যেমন ব্লক ডিভাইসগুলোকে নিজের আভ্যন্তরীন করে, কিন্তু ক্যারেকটার

ডিভাইসগুলোকে ছেড়ে দেয় আরো নিচের স্তরের সফটওয়ারদের হাতে ছেড়ে দেয় — আমরা পরে আসব এই নিয়ে দীর্ঘ আলোচনায়।

ডিভাইস	তথ্যহার/সেকেন্ড
কিবোর্ড	১০ বাইট
মাউস	১০০ বাইট
৫৬-কে মোডেম	৭ কেবি
টেলিফোন লাইন	৮ কেবি
লেজার প্রিন্টার	১০০ কেবি
স্ক্যানার	৪০০ কেবি
ইথারনেট	১.২৫ এমবি
আইডিই ডিস্ক	৫ এমবি
৪০ x সিডিরম	৬ এমবি

আইও নিয়ে আমাদের এই আলোচনাটা আপাতত শেষ হল, বিভিন্ন ডিভাইসের তথ্য দেওয়া-নেওয়া হারের একটা তালিকা দিয়ে। অপারেটিং সিস্টেমকে তৈরি থাকতে হয় এই আলাদা আলাদা গতিবেগগুলোর প্রত্যেকটার সঙ্গে মানিয়ে নেওয়ার। বিভিন্ন আইও ডিভাইসের এবং বাস-সংযোগের এই বিট-রেট বা তথ্য বহন করার হার — এটা কিন্তু রোজ বদলাচ্ছে। এটা আমি তুলছি জুলাই ২০০২-এ ছাপা অ্যান্ড্রু ট্যানেনবমের অপারেটিং সিস্টেম বইটা থেকে। সায়মিন্দু কাল প্রস্তাব দিল, যে এই পাঠমালাটা অঙ্কুর বাংলা-লাইভ সিডিতে রাখবে — তার মানে আপনার পড়তে পড়তে আরামসে ২০০৪ হয়ে যাচ্ছে — এর মধ্যে কোথাকার বিট কোথায় গড়াবে কে জানে।

৭।। আইও তথ্য, ইন্টেরাপ্ট, ডিএমএ

প্রতিটি কন্ট্রোলারের মধ্যে থাকে কয়েকটা করে রেজিস্টার। রেজিস্টার তো আমরা চিনি, সিপিইউ-র মধ্যে থাকে যেমন, এক নম্বর দিনে আলোচনা হয়েছে। অপারেটিং সিস্টেম যখন কোনো আইও ডিভাইস নিয়ে কিছু করতে চায়, তাকে কথা বলতে হয় এই ডিভাইস কন্ট্রোলারের সঙ্গে আর সেই কথোপকথনের দোভাষীর কাজ করে এই রেজিস্টারগুলো। একটা হার্ডডিস্ক কন্ট্রোলারে যেমন সচরাচর রেজিস্টার থাকে ডিস্কের শরীরে ব্লক-ঠিকানা চিহ্নিত করার, মেমরির শরীরে বাইট-ঠিকানা চিহ্নিত করার (এক নম্বর দিনের কূট কিন্তু আপাতঅগ্রাহ্য সেকশন থেকে একটু দেখে নিন মেমরি ঠিকানার কনসেপ্টটা, যদি ভুলে গিয়ে থাকেন), সেক্টর গোনার, এবং তথ্যের গতিমুখের — মানে ঢুকছে না বেরোচ্ছে, তথ্য লেখা হচ্ছে না পড়া হচ্ছে, ইত্যাদি। আমরা আগেই বলেছি, ডিভাইসকে হার্ডওয়ার নিয়ন্ত্রণ করে কন্ট্রোলার, আর এই কন্ট্রোলারকে তথা ডিভাইসকে সফটওয়ার নিয়ন্ত্রণ করার জন্যে থাকে একটা ডিভাইস ড্রাইভার যা কারনেল বা অপারেটিং সিস্টেমের সঙ্গে যুক্ত হয়ে যায়, যাতে অপারেটিং সিস্টেম ওই ডিভাইসের রসদটাকে বন্টন করতে পারে অ্যাপ্লিকেশন প্রোগ্রামগুলোর ভিতরে। এবার, যখন অপারেটিং সিস্টেম একটা ডিভাইসকে কাজে লাগাতে চায়, একটা সিগনাল পাঠায় এই ডিভাইস ড্রাইভারের কাছে। এই সিগনালকে তখন কন্ট্রোলারের কাছে পাঠিয়ে দেয় ডিভাইস ড্রাইভার ওই রেজিস্টারগুলোর মারফত।

একদিকে সিপিইউ, আর অন্যদিকে একদম ভৌত স্তরের ডিভাইস — এই দুই মেরুর মধ্যে চলাচল করে তথ্য, আইও বা ইনপুট আউটপুট তথ্য। সেটাই কাজ আইও ডিভাইসগুলোর, বাইরের তথ্য সিপিইউতে আনা, এবং সিপিইউ-র তথ্য বাইরে নিয়ে যাওয়া। এই তথ্য যাতায়াতের কাজটা মূলত ঘটে তিন প্রক্রিয়ায়। তিনটে পদ্ধতির নাম হল ব্যস্ত-অপেক্ষা বা বিজি-ওয়েটিং, ইন্টেরাপ্ট বা ব্যাঘাত, এবং ডিএমএ (DMA — **D**irect-**M**emory-**A**ccess) বা সরাসরি-স্মৃতি-সংযোগ। একটু আলতো করে ছুঁয়ে আসা যাক এই তিনটে আইও তথ্য যাতায়াতের রকমকে। এর মধ্যে ইন্টেরাপ্ট আর ডিএমএ-র প্রসঙ্গ গোটা পাঠমালা জুড়েই বারংবার আসবে।

৭.১।। ব্যস্ত-অপেক্ষা বা বিজি-ওয়েটিং

আইও তথ্য যাতায়াতের সবচেয়ে সরল পদ্ধতি হল বিজি-ওয়েটিং। ইউজারের চালানো একটা প্রয়োগ বা অ্যাপ্লিকেশন সফটওয়ার, তার কাজের সূত্রে, কিছু পরিমাণ তথ্য নড়াতে চাইল এক জায়গা থেকে আর এক জায়গা। ডিভাইস থেকে ফাইল পড়া হতে পারে, ডিভাইসে ফাইল লেখা হতে পারে, ডিভাইস-মেমরি-সিপিইউ এদের মধ্যে কাজগত নাড়াচাড়াও হতে পারে। প্রয়োজন পড়া মানে, অ্যাপ্লিকেশন প্রোগ্রাম এবার ওএস মানে কারনেল মানে চূড়ান্ত রসদ-মঞ্জুরি কমিশনের কাছে আর্জি জানাল। নিজে নিজে রসদ উপাদান ব্যবহার করে নেওয়ার তো তার অধিকার নেই, আগেই বলেছি। এবার, অ্যাপ্লিকেশন প্রোগ্রাম ওএস বা কারনেলের কাছে তার রসদ ব্যবহারের আর্জি জানানো মাত্র, এই আর্জিটার চালু টেকনিকাল নাম হল সিস্টেম-কল (system-call), সেই সিস্টেম-কলটাকে কারনেল অনুবাদ করে নিল একটা আদেশে, যা ডিভাইস ড্রাইভারের কাছে কারনেল পাঠাবে। ডিভাইস ড্রাইভারের কাছে কারনেলের পাঠানো এই আদেশটার টেকনিকাল নাম হল প্রসিডিওর কল (procedure-call)।

ওএস-এর কাছ থেকে প্রসিডিওর কল আসা মাত্র ড্রাইভার এবার চালু করে তোলে ওই ডিভাইসকে তথা ডিভাইসের মাধ্যমে আইও তথ্য পড়া বা লেখার কাজ, এবং নির্নিমেষ পর্যবেক্ষণ করে চলে সক্রিয় ডিভাইসটাকে। আর বাধ্য ডিভাইস তার ড্রাইভারের আদেশানুযায়ী আইও তথ্যের কাজ করে চলে। যেই আইও তথ্য পড়া/লেখার কাজটা শেষ হয়, ড্রাইভার তখন প্রসিডিওর কল মোতাবেক তার যেখানে যেখানে তথ্য পাঠানোর আছে — সেই গোটা কাজটা শেষ করে ফেরত আসে তার স্বাভাবিক নিষ্ক্রিয় অবস্থায়। ওএস বা কারনেলও ফেরত যায় অ্যাপ্লিকেশন প্রোগ্রাম থেকে সিস্টেম কল আসার আগের পুরোনো অবস্থায়। এই সিস্টেম-কল থেকে প্রসিডিওর-কল থেকে ডিভাইস ড্রাইভার তথা ডিভাইসের সক্রিয়তা থেকে পর্যবেক্ষণ থেকে ফের পুরোনো নিষ্ক্রিয়তা — এই গোটা লুপটা শেষ হয়। এটাই ব্যস্ত-অপেক্ষা বা বিজি-ওয়েটিং পদ্ধতি।

পর্যবেক্ষণটা ঘটে একটা লুপ বা পুনরাবৃত্ত পথে — এক নম্বর দিনের ওই fetch … repeat লুপটা মনে করুন, অনেকটা ওই রকম। দেখে, বসে থাকে, দেখে . . . এইরকম চলতেই থাকে। যতক্ষণ না ওই ইনপুট/আউটপুটের কাজ শেষ হয়। যেই শেষ হয় তখনি ড্রাইভার তথ্যটাকে যেখানে দরকার সেখানে পাঠিয়ে দেয় এবং ফের নিষ্ক্রিয় হয়ে যায়। অপারেটিং সিস্টেম আবার ইউজার প্রোগ্রামটাকে জানিয়ে দেয়, তোমার কাজ শেষ হয়ে গেছে। এই কায়দার অসুবিধেটা হল এই যে কাজের গোটা সময়টা জুড়ে ওই লুপে আটকে থাকতে হয় ডিভাইস কন্ট্রোলারকে, আর তাই আটকে থাকতে হয় ডিভাইস ড্রাইভার দিয়ে কন্ট্রোলারকে যে চালাচ্ছে সেই সিপিইউ বা প্রসেসরকেও। সেই অর্থে এই পদ্ধতিটা তাই ইনএফিশিয়েন্ট বা ক্যাবলা, একটা কম্পিউটারের সবচেয়ে দামী এবং জরুরি জিনিষটাকে আটকে রাখে তার খ্যাঁচাকলে।

৭.২।। ইন্টারাপ্ট বা ব্যাঘাত পদ্ধতি

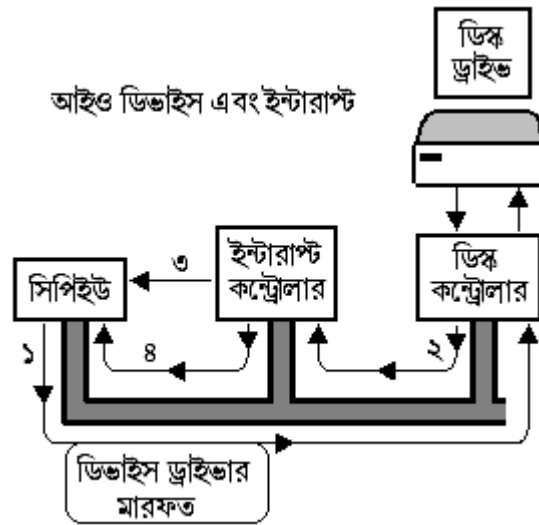
দু-নম্বর কায়দাটায় অ্যাপ্লিকেশন প্রোগ্রামের সিস্টেম-কল কারনেলের মাধ্যমে প্রসিডিওর কলে পরিণত হয়ে ডিভাইস ড্রাইভারের কাছে যায়। ডিভাইস ড্রাইভার ডিভাইসকে চালু করে দেয়, করে জানিয়ে দেয়, কাজ শেষ হলে ভাই আমাকে একটা ফোন করে দিস। এই ফোনটাই হল ইন্টেরাপ্ট। ডিভাইসটার ভৌত শরীরে ঘটমানতার বিষয়ে ডিভাইস ড্রাইভারের কারনেলের কাছে পাঠানো সিগনাল। এই সিগনালটা পাঠিয়ে দিয়ে ডিভাইস কন্ট্রোলার ফেরত চলে আসে তার পুরোনো অবস্থায়, কারনেলের কাছ থেকে ডিভাইস ড্রাইভার মারফত নতুনতর আদেশ আসা অব্দি ঘুমোতে থাকে। অপারেটিং সিস্টেম এবার যে ইউজার প্রোগ্রাম রসদ ব্যবহারের ওই আর্জিটা জানিয়েছিল তার কাছ থেকে নতুনতর কোনো আর্জি আসার পথটা আপাতত বন্ধ করে দিয়ে চোস্ত পাজামা আর গোড়ালি অব্দি লম্বা পাঞ্জাবি পরে রাজনৈতিক রোঁদে বেরোয় — আর কোনো প্রোগ্রামের কোনো আর্জি আছে কিনা, দেখতে হবেনা জনগণ কেমন আছে। আর কারুর জন্যে কিছু করতে হবে ভাই? ডিভাইস কন্ট্রোলার ওদিকে তার ডিভাইসের কাজ শেষ হলে যথারীতি কথামতো একটা ফোন করে দেয় ড্রাইভারের কাছে — ইন্টেরাপ্ট। কারনেলের কাছে খবর যায়, ওদিকটা ফ্রি হয়ে গেছে দাদা। আবার কোনো আর্জি নিতে পারো। এই প্রোগ্রামের হোক, বা অন্য প্রোগ্রামের।

ওএস বা অপারেটিং সিস্টেমের কার্যপ্রনালীতে ইন্টারাপ্ট বা ব্যাঘাত একটা জরুরি ভূমিকা পালন করে। ছবিতে দেখুন, আমরা তিনটে আবশ্যিক এবং একটা সম্ভাব্য ধাপে ভেঙে গোটা আইও তথ্য চলাচল এবং ইন্টারাপ্টের পুরো পদ্ধতিটা দেখিয়েছি।

ধাপ ১।

সিপিইউ ডিভাইস ড্রাইভার মারফত তার আদেশ পাঠাচ্ছে ডিস্ক কন্ট্রোলারকে। মানে ডিভাইস কন্ট্রোলারের রেজিস্টারগুলোতে দেগে দিচ্ছে — তার ঠিক কী চাই। কন্ট্রোলার এবার চালু করে দিচ্ছে ভৌত ডিভাইসটাকে। সে একদম ভৌত স্তরে তার তথ্য লেখার বা পড়ার কাজে ব্যস্ত হয়ে যাচ্ছে।

ধাপ ২।

কন্ট্রোলার গোটা সময়টা জুড়ে তার ঘনিষ্ঠ পাহারাটা চালিয়ে যাচ্ছে ডিভাইসের উপর। যেই, সিপিইউর আদেশমত বাইট স্থানান্তরের গোটা কাজটা সাঙ্গ হয়ে যাচ্ছে, কন্ট্রোলার সেই সংবাদটা জানিয়ে দিচ্ছে ইন্টেরাপ্ট কন্ট্রোলার চিপকে। ঠিক ডিভাইস গুলোর যেমন কন্ট্রোলার চিপ এবং সার্কিট থাকে, ইন্টারাপ্ট বা ব্যাঘাতের গোটাটা নিয়ন্ত্রণ করার জন্যেও একটা নিয়ন্ত্রক সার্কিট থাকে। ডিভাইস কন্ট্রোলার এই সংবাদটা ইন্টারাপ্ট কন্ট্রোলারকে পাঠাচ্ছে তথ্য চলাচলের জন্যে কম্পিউটার জুড়ে ছড়ানো যোগাযোগপথ বাসের ভিতরেই বিশেষ কিছু লাইন বেয়ে।

ধাপ ৩।

ইন্টেরাপ্ট কন্ট্রোলার কিন্তু সেই মুহূর্তে ইন্টেরাপ্ট নেওয়ার অবস্থায় থাকতেও পারে, আবার নাও পারে, যদি তার হাতে আরো জরুরি কোনো ইন্টেরাপ্ট থাকে। এই মুহূর্তে যদি তার ইন্টেরাপ্ট নেওয়ার অবস্থা থাকে, সে ইন্টেরাপ্টটা গ্রহণ করে এবং ইন্টেরাপ্টের ভিতরে থাকা আইও তথ্য স্থানান্তরের কাজ শেষ হওয়ার সিগনালটা পাঠিয়ে দেয় সিপিইউর কাছে।

ধাপ ৪।

এই ধাপটা তখনই ঘটে, যদি একাধিক আইও ডিভাইস একই সঙ্গে চালু অবস্থায় থাকে। তখন ইন্টেরাপ্ট কন্ট্রোলার যে ডিভাইসের তথ্য স্থানান্তরের কাজ হল, সেই ডিভাইসের নম্বরটা বাসে তুলে দেয়, যাতে সেখান থেকে পড়ে কারনেল কোন আইও ডিভাইস ফাঁকা হয়েছে। এই নম্বর দিয়েই কারনেল একটা ডিভাইসকে চেনে। ছয় নম্বর দিনে আমরা এই ডিভাইস এবং তার নম্বরের বিষয়টা জানব, ডিভাইস ফাইলের সূত্রে। আমরা দেখব, গ্নু-লিনাক্সে সবকিছুই ফাইল, এমনকী ডিভাইসগুলোও।

ডিএমএ বা সরাসরি-স্মৃতি-সংযোগ পদ্ধতি

আইওর তৃতীয় কায়দাটা কাজ করে একটা ডিএমএ চিপ (DMA — **D**irect-**M**emory-**A**ccess) মারফত। এই ডিএমএ চিপটা সরাসরি ডিভাইস কন্ট্রোলার আর মেমরির মধ্যে তথ্যের যাতায়াত নিয়ন্ত্রণ করে। আইও তথ্যের স্থানান্তরের উপর অহর্নিশ বাধ্যতামূলক সিপিইউ-র খবরদারিটা আর ঘটেনা এই পদ্ধতিতে। এখানে সিপিইউ ডিএমএ চিপকে জানিয়ে দেয় — কতগুলো বাইট নড়াতে হবে, ডিভাইসের ঠিকানা কী, মেমরির ঠিকানা কী, এবং কোনদিকে যাবে তথ্যটা, ডিভাইস থেকে মেমরি না মেমরি থেকে ডিভাইস। ডিএমএ চিপের কাজ যেই শেষ হয়, সে অমনি একটা ইন্টেরাপ্ট পাঠায়। অপারেটিং সিস্টেম টের পায় কাজ শেষ হয়ে গেছে।

অপারেটিং সিস্টেমের কাছে পাঠানো এই ইন্টেরাপ্টগুলোকে নাড়াচাড়া করার জন্যে যে ইন্টেরাপ্ট কন্ট্রোলারটা থাকে, সে কোনো একটা মুহূর্তে কোনো একটা ডিভাইস থেকে আসা ইন্টেরাপ্টকে চাইলে স্থগিত করে দিতে পারে, বা নতুন করে সক্রিয় করে তুলতে পারে, কারণ, প্রায়ই ইন্টেরাপ্ট কন্ট্রোলারের কাছে একই সঙ্গে একাধিক ইন্টেরাপ্ট এসে পড়ে। তখন কোন ইন্টেরাপ্টকে আগে মনোযোগ দেবে ইন্টেরাপ্ট কন্ট্রোলার সেটা ঠিক হয় প্রায়োরিটি বা প্রাথমিকতা দিয়ে। কোন ইন্টেরাপ্টটা বেশি জরুরি। পরে, যখন প্রসেস এবং প্রায়োরিটি নিয়ে, পদ্ধতি এবং তার প্রাথমিকতা নিয়ে আমরা আলোচনা করব তখন এই ধারণাটা আমাদের দরকার পড়বে। আর এই সেকশনেই আমরা যে কথাগুলো বলেছিলাম মাল্টিপ্লেক্সিং নিয়ে, পরে আরো বলব, তার সঙ্গে মিলিয়ে নিন একে। এই কালগত মাল্টিপ্লেক্সিং, ভূমিগত মাল্টিপ্লেক্সিং, এই লাইনেই আসবে টাইমশেয়ারিং, মাল্টিপ্রোগ্রামিং, মাল্টিটাস্কিং-এর আলোচনা। চার এবং পাঁচ নম্বর দিনে।

৮।। বাস-যোগাযোগ

আজকের আলোচনারই একদম গোড়ায় আমরা একটা বাস-যোগাযোগব্যবস্থার ছবি দেখিয়েছি। এটা হল একটা সরল বাস। এখনকার বাস এর চেয়ে জটিল হয়। এই সহজতর বাস ব্যবহৃত হত একদম গোড়ার দিককার আইবিএম পিসিতে। পাঁচ নম্বর দিনে এ নিয়ে আরো বিশদ আলোচনা আসবে, তবু মূল কথাটা এই যে ক্রমে প্রসেসরগুলোর গতিবেগ বাড়ছিল, মেমরিদেরও। এই নতুন ধরনের প্রসেসর আর মেমরি কাজে লাগিয়ে তথ্যের ট্রাফিকের মোট পরিমাণটাও বাড়ছিল হুহু করে। তার সঙ্গে আর পাল্লা দিতে পারছিলনা এই সরল এক রাস্তার বাসব্যবস্থা। তাই এর সঙ্গে আরো আরো বাস যোগ করা হচ্ছিল। একদিকে সিপিইউ আর মেমরির মধ্যে। অন্যদিকে নতুন যুগের দ্রুততর আইও ডিভাইসগুলোর সঙ্গে তাদের কন্ট্রোলারদের তথা সিপিইউর। তবে তাদের আলোচনা করতে গেলে যে পরিমাণ হার্ডওয়ার আলোচনা আনতে হবে সেটা খুব একটা পছন্দ হচ্ছেনা। আর সেগুলো বোঝানো মানে নতুন ছবি আঁকা, আর ডাক্তারের আদেশ মোতাবেক আমাকে বসতে হচ্ছে ডান পাটা উপরে তুলে, টেবিলের পাশে খাটে একটা জলচৌকি রেখে। এই অবস্থায় মাউস চালাতে অসুবিধে হচ্ছে খুব। কোমরে লাগছে। ডাক্তার বলেছে ভেঙেছে পায়ের পাতার মেটাটার্সাল বা এই গোছের দুটো হাড়। অথচ ব্যাথা করছে গোড়ালি হাঁটু হয়ে কোমর অব্দি। আমার পায়ের পাতাটা যে আমার কোমর অব্দি এটা আমার আগে জানা ছিলনা। এত ব্যথাট্যথা নিয়ে বাসজার্নি ভালো না। ব্যস, বাস এখানেই শেষ। এর পরে সিস্টেম লিখতে গিয়ে খুব প্রয়োজন পড়লে একটু ছোট করে মেরে দেওয়া যাবে। পা ভাঙল বুধবার, বারোই নভেম্বর রাতে। আজ রবিবার। সোয়া চার দিন গেছে, এর মধ্যে এক আর দুই নেমে গেল। অবশ্য প্রথম খসড়া তো ছিলই। জয় পা।

আমাদের এই গ্নু-লিনাক্স ইশকুলের পাঠটাও কাজ করে চলেছে একের পর এক ফাইল জুড়ে, জিএলটি-এলইএস০০, জিএলটি-এলইএস০১, জিএলটি-এলইএস০২, ... কিন্তু সেটা তো শুধু মৃত স্থির পরিবর্তনহীন ফাইল — গতিশীল ফাইল — প্রোগ্রাম থেকে প্রসেস, তাই ফাইল-আবদ্ধ তার আকারটাও এই জ্যান্ত গতির প্রকোপে বদলে যাচ্ছে — গ্নু-লিনাক্সে ভাবতে শেখার সাহায্যের জন্যে লেখা জিএলটি ইশকুল পাঠমালাটা নিজেও জ্যান্ত। আমরা তার জ্যান্ত অংশীদার। তাই ফাইলে ফাইলে দেগে দেওয়া আমাদের স্থির পরিকল্পনাটাও অস্থির হচ্ছে, নিয়ত বদলাচ্ছে। এখনো আমরা অপারেটিং সিস্টেমের ইতিহাসে ঢুকলাম না।

এখনো আমাদের মূল রেফারেন্স ট্যানেনবম-এর মডার্ন অপারেটিং সিস্টেম আর জিএলটির থেকে দেওয়া গ্নু-লিনাক্স রিসোর্স সিডির লিনাক্স ডিকশনারি এবং রেডহ্যাট লিনাক্সের ডকুমেন্টেশনে গ্লসারিটা। এছাড়া আর একটু ভালো করে বাস ইত্যাদি গুলো বুঝতে চাইলে থম্পসন অ্যান্ড থম্পসন-এর হার্ডওয়ার ইন এ নাটশেল তো আছেই। ভালো কথা, আগের দিন যে সিপ্লাসপ্লাস-এর বইয়ের নামটা মনে পড়েনি সেটার নামটা পরে নেটে পেয়েছি। নিল গ্রে-র এ বিগিনার্স সিপ্লাসপ্লাস। অত্যন্ত উমদা বই। ওরিলির। অপারেটিং সিস্টেম আর প্রোগ্রামিং তথা প্রোগ্রামিং ল্যাংগোয়েজ-এর পারস্পরিক যে সম্পর্কটা — সিস্টেম কল, প্রসিডিওর কল — এগুলো খুব ভালো বোঝানো আছে। আর একটা বইতেও খুব ভালো করে আছে, স্টিফেন প্রাটার ইউনিক্স অ্যাডভান্সড প্রোগ্রামিং। পরের বইটার প্রসঙ্গ পরে খুব বেশি করে আসবে, এই পাঠমালার শেষ মানে দশ নম্বর দিনে — যখন আমরা ব্যাশ তথা শেল স্ক্রিপ্ট নিয়ে কথা বলব।

ও, একটা জিনিষ, আমাদের লাগের ইন্দ্র আর সঙ্কর্ষণ — ওদের দুজনেরই ওই কম্পিউটারকে গাধা বলে ডাকার জন্যে লেখা প্রোগ্রামটা পছন্দ হয়নি খুব একটা। একটু বোকাবোকা লেগেছে বোধহয়। ইন্দ্রর সঙ্গে এখনো সামনাসামনি কথা হয়নি, সঙ্কর্ষণ নিজেই বলেছে। আপনাদের যদি এই নিয়ে কিছু বলার থাকে, বা অন্য কিছু নিয়েও, জানান। মতামতের জন্যে অপেক্ষা করে আছি, আরো সেই মতামত যদি আমারে ফেভারে যায়। জিএলটির মেল আইডি তো দেওয়াই আছে, প্রতিটি দিনের শেষে জিএলটির লোগোয়। আর সায়মিন্দু দিন এক-এর একটা ভুল ধরিয়ে দিয়েছে — ল্যান কার্ড কথাটা বলে চলে গেছি, কিন্তু এই অ্যাক্রোনিমটার পিছনে গোটা নামটা দিইনি। লোকাল এরিয়া নেটওয়ার্ক। এই কার্ড দিয়ে কম্পিউটার অন্য একটা ল্যানিত কম্পিউটারের সঙ্গে কথা বলে।

সংকলন ও রচনা : মধ্যমগ্রাম জিএলটি-র (glt-mad@ilug-cal.org) তরফে ত্রিদিব সেনগুপ্ত